UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

TRABAJO FIN DE MÁSTER

GENERACIÓN AUTOMÁTICA DE CHATBOTS PARA LA CONSULTA DE DATOS ABIERTOS

Máster Universitario en Ingeniería Informática

Sandra Juárez Puerta

Tutor: Juan de Lara Jaramillo

Tutora: Esther Guerra Sánchez

Departamento de Ingeniería Informática

3 de septiembre de 2020

GENERACIÓN AUTOMÁTICA DE CHATBOTS PARA LA CONSULTA DE DATOS ABIERTOS

Autor: Sandra Juárez Puerta

Directores: Juan de Lara Jaramillo

Esther Guerra Sánchez

Grupo de Modelado e Ingeniería del Software (MISO)

Departamento de Ingeniería Informática

Escuela Politécnica Superior

Universidad Autónoma de Madrid

17 de febrero de 2020

Resumen

A día de hoy, con el auge de las nuevas tecnologías y la automatización de prácticamente todas las tareas cotidianas, se recopilan grandes cantidades de datos. Numerosas entidades públicas y privadas se suman a un movimiento que apuesta por la transparencia y libertad de la ciudadanía para manipular y consultar todos los datos que generan. Así, nacen los datos abiertos, fuentes de información que se exponen libremente para su utilización y distribución [1, 2].

Para obtener información de estos conjuntos de datos, a menudo se usan entornos y lenguajes de consulta técnicos no accesibles para todos los usuarios. Sin embargo, no todas las personas tienen las cualidades técnicas necesarias para trabajar con ellos.

Así nace GENBOT, un programa informático con el que se interactúa a través del lenguaje natural (chatbot) en un entorno conocido, en este caso, una aplicación de mensajería como Telegram. Debido a la gran y diversa cantidad de información abierta disponible, GENBOT tiene por objetivo generar automáticamente chatbots de consulta y visualización personalizados ligados a fuentes de datos concretas. Específicamente, en este proyecto se trabajará con fuentes de datos abiertas en formato CSV.

Con estos chatbots autogenerados, el usuario, a través del lenguaje natural podrá realizar consultas y visualizar datos de la fuente vinculada. En concreto, para el procesado del lenguaje natural de estos chatbots y de GENBOT, se usará Dialogflow, un framework de creación de bots con un motor de inteligencia artificial (IA).

Finalmente, este proyecto ha sido probado en su totalidad a través de dos casos de uso con datos abiertos de la Comunidad de Madrid en formato CSV.

Palabras clave – Chatbot, datos abiertos, lenguaje natural, SQL, aplicaciones de mensajería, consultas, visualización de datos.

Abstract

Today, with the rise of new technologies and the automation of every day job, large amounts of data are collected. Many public and private entities have joined a movement that is committed to transparency and freedom of citizens to manipulate and consult all the data they generate. This is called "open data", data that are freely exposed for use and distribution (1; 2).

To obtain information from these datasets, technical query languages and environments are often used but they are not accessible to all users. Moreover, not all people have the technical qualities necessary to work with them. This is how GENBOT was born, a computer program with which the users interact through natural language (chatbot) in a familiar environment, in this case, a messaging application like Telegram.

Due to the large and diverse amount of open data available, GENBOT aims to automatically generate custom query chatbots linked to specific datasets. Specifically, this project will work with datasets in CSV format.

With these self-generated chatbots, the user, through natural language, can make queries and view data from the linked source. Specifically, for the natural language processing of these chatbots and GENBOT, Dialogflow will be used. It is a framework for creating bots with an artificial intelligence engine.

Finally, this project has been fully tested through two use cases with open data from the Community of Madrid in CSV format.

Key words – Chatbot, open data, natural language, SQL, messaging applications, queries, data visualization.

Contenido

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

En este capítulo se proporciona una visión general del proyecto (Sección 1.1) junto con la motivación para llevarlo a cabo (Sección 1.2). Además, se exponen los objetivos que quieren satisfacerse con la realización de este proyecto (Sección 1.3) junto con una breve descripción de todos los capítulos que conforman este documento (Sección 1.4). En la última sección se encuentra cómo se ha financiado este proyecto (Sección 1.5).

1.1 Visión general

El objetivo fundamental de este sistema es el de garantizar el acceso a datos abiertos para consulta y visualización de información a usuarios sin conocimientos previos en el tratamiento de datos. De esta manera, el usuario podrá recopilar información tras realizar consultas en lenguaje natural a un chatbot, ligado a una fuente de datos, a través de una aplicación de mensajería.

Debido a todas las fuentes de datos abiertas existentes, es imposible pensar en la implementación de un chatbot genérico válido para todas ellas. Es por tanto que, para cada fuente, se podrá crear de manera automatizada un chatbot distinto con un reconocimiento de semántica específica para cada conjunto de datos. En concreto se trabajará con fuentes de datos en CSV, un formato fácil de usar y que está bastante extendido tanto en el ámbito de los datos abiertos como en otros.

El sistema que se describe en este documento se centra en dos puntos distintos. Uno de ellos es la diversidad de los datos abiertos, es decir, cada conjunto de datos tiene una estructura distinta, mientras que el segundo es facilitar el acceso a estos datos a cualquier tipo de usuario. De estos dos puntos nace este proyecto: acercar cualquier conjunto de datos, dentro del formato soportado, a cualquier usuario.

1.2 Motivación

A día de hoy, la digitalización y la automatización han llegado a cada una de nuestras actividades y tareas; esto ha producido la necesidad de almacenar, tratar y consultar una cantidad ingente de datos con un gran valor en cualquier ámbito que se pueda imaginar.

Debido a la gran cantidad de datos que se recogen, ha nacido un nuevo concepto de almacenamiento y difusión que se centra en hacer accesibles, transparentes y reutilizables todos estos datos sin exigir permisos específicos. Así, nacen las fuentes de datos abiertas [3].

Multitud de organizaciones, tanto públicas como privadas, así como científicas, contribuyen a esta causa y, a su vez, a la sociedad con la publicación en abierto de todos los datos que generan.

Sin embargo, a pesar de que cualquier persona puede acceder a ellos, éstos resultan difíciles de manejar para usuarios inexpertos en el tratamiento de datos. Esta dificultad radica en que el volumen de datos es, a veces, muy elevado y, además, no tienen una estructura homogénea, ni una interfaz amigable para su consulta y manipulación.

El proyecto que se presenta, pretende facilitar la consulta y visualización de información de fuentes de datos abiertas a cualquier tipo de usuarios. Para lograr este acercamiento y mejorar al máximo la experiencia de usuario, creando un sistema lo más usable y fácil de manejar por cualquier persona [4], usaremos el lenguaje natural.

Desde hace numerosos años, existen programas informáticos con los que el usuario puede comunicarse en lenguaje natural, son los llamados chatbots [5]. Concretamente, grandes organizaciones como Google, Microsoft, Amazon o IBM han desarrollado frameworks específicos con procesamiento del lenguaje natural a través de Inteligencia Artificial que pueden usarse para la creación de chatbots de forma sencilla. Así, este proyecto se beneficiará de las facilidades que ofrecen estos frameworks para la creación chatbots, en concreto se trabajará con Dialogflow de Google.

A su vez, debido a la gran cantidad de conjuntos de datos con estructuras muy distintas entre sí, se hace muy difícil poder crear un bot de consulta genérico válido para todas estas fuentes. Por ello, en este proyecto, se propone la creación de un chatbot único adaptado a cada conjunto de datos. Este chatbot, a su vez,

se creará de manera automatizada a través de otro, un "super" chatbot llamado GENBOT.

Para la comunicación con los chatbots, el usuario no tendrá que manejar nuevas aplicaciones o dispositivos, si no que podrá usar aquellas plataformas a las que ya está acostumbrado. Hablamos de las aplicaciones de mensajería instantánea, las cuales pueden usarse a través de distintos dispositivos como PCs o teléfonos móviles. Se aprovechará la familiaridad de uso de estas aplicaciones para integrar en ellas chatbots con los que el usuario podrá establecer una conversación. En concreto, se usará Telegram ya que es una de las aplicaciones de mensajería con soporte para bots más extendidas y usadas.

Además, todo el intercambio de mensajes que se produzcan entre el usuario y el bot, mediante Telegram, no tiene por qué ser necesariamente texto, sino que, se puede aprovechar toda la riqueza de opciones de visualización e interación que ofrece. Así, la conversación será más interesante, eficiente y visual [5]. Por ejemplo, cuando hay que elegir entre distintas opciones (ya predefinidas) se puede implementar el uso de botones en Telegram que proporciona una interacción del usuario mucho más eficaz e intuitiva.

En resumen, el proyecto que se presenta en este documento tiene la finalidad de facilitar la consulta y visualización de información de fuentes de datos abiertas a cualquier tipo de usuarios. Por tanto, a través de GENBOT, se podrá obtener un chatbot que se integrará en Telegram e interpretará consultas en lenguaje natural que después, serán traducidas a un lenguaje específico (SQL) para tratar con las fuentes de datos especificadas.

1.3 Objetivos

El objetivo general que se persigue con la consecución de este proyecto es el de realizar consultas de manera sencilla, usando el lenguaje natural, a través de una aplicación de mensajería a las fuentes de datos abiertos que se especifiquen.

Como objetivos más concretos encontramos:

1. Acercar usuarios inexpertos al acceso a datos abiertos.

2. Acceder al servicio desde una interfaz conocida por los usuarios (aplicaciones de mensajería) a través del lenguaje natural.

3. Automatizar la creación de un chatbot específico para cada fuente de datos dada.

4. Consultar información y visualizar mapas y gráficas dado un conjunto de datos.

1.4 Estructura del documento

La estructura del documento sigue el siguiente esquema, donde se describirán, a continuación, cada uno de los capítulos de los que consta.

- Capítulo 2. Estado del arte. En este capítulo se encuentra una descripción de las tecnologías y herramientas relacionadas con el proyecto junto con una introducción al entorno donde se enmarca el sistema realizado.
- Capítulo 3. Enfoque expone una visión general del funcionamiento del sistema.
- Capítulo 4. Arquitectura y herramienta explica en detalle el funcionamiento y arquitectura del sistema.
- Capítulo 5. Caso de uso y pruebas. En este capítulo se detallan dos casos de uso reales que prueban el sistema en su totalidad.
- Capítulo 6. Conclusiones y trabajo futuro contiene las conclusiones que nacen a partir de la realización del proyecto y el trabajo futuro.

1.5 Financiación

La realización de este proyecto ha sido financiada por el Fondo Social Europeo dentro del Programa Operativo de Empleo Juvenil de la Comunidad de Madrid (PEJD-2018-PRE/BMD-8667). Además, ha dado lugar a la siguiente publicación que está en fase de evaluación:

Sara Pérez Soler, Sandra Juárez Puerta, Esther Guerra, Juan de Lara. "Choosing a chatbot development tool". 2020 (en evaluación). IEEE Software (JCR 2.945, Q1 Software Engineering)

Capítulo 2
Antecedentes y estado del arte

Este capítulo tiene como objetivo exponer las distintas plataformas y tecnologías que se han usado y evaluado de acuerdo a crear el sistema descrito y describir el trabajo relacionado.

En la sección 2.1 se puede encontrar una breve explicación para entender qué son y para qué se usan los chatbots. Además, también se expondrá un breve resumen de las plataformas para el desarrollo de bots más importantes (Sección 2.1.1).

Después, en la sección 2.2 se describirán las aplicaciones de mensajería evaluadas con las que el usuario podría interactuar con el chatbot propuesto.

Finalmente, en el apartado 2.3 se comparará este proyecto con trabajos similares.

2.1 ¿Qué es un chatbot?

Un chatbot es un programa informático con el que un usuario puede mantener una conversación en lenguaje natural. Comúnmente, el fin de estas plataformas es el de ofrecer al usuario información, usar servicios concretos o servir como entretenimiento [5]. Por ejemplo, los servicios más populares que se implementan con chatbots son aquellos para hacer reservas de hotel o avión, comprar o pedir ayuda sobre productos en páginas webs. En general, suelen enfocarse a la atención al cliente.

Hace unos años, el usuario tenía que adaptarse a un entorno técnico específico para poder obtener un servicio concreto. Ahora, la importancia y el valor de la experiencia de usuario en las aplicaciones, junto con el avance de la tecnología e Inteligencia Artificial, han popularizado el uso de chatbots como sistemas intermediarios que ofrecen servicios usando el lenguaje natural [6, 5]. Así ahora, por ejemplo, una persona podrá comprar un billete de avión por medio de una simple frase, “quiero un billete de avión a Londres mañana”, en lugar de entender cómo funciona la aplicación específica de compra de billetes de avión de la aerolínea elegida.

Existen multitud de tipos de chatbots distintos. Entre ellos podemos distinguir: los que hacen uso de IA para interpretar los mensajes entrantes, los que se basan en el pattern-matching para categorizar mensajes y los que siguen un flujo concreto y estático de diálogo a través de opciones o esperando respuestas del usuario bien definidas.

En cuanto a herramientas para la creación de bots basadas en pattern-matching, podemos encontrar frameworks como Botkit [7] o plataformas como FlowXo [8], Chatfuel [9] y SmartLoop [10]. Tales herramientas se enfocan en detectar patrones previamente definidos en los mensajes del usuario y brindar, así, las respuestas adecuadas a cada mensaje. Por otro lado, con Landbot [11], podemos crear potentes bots con estructuras de diálogo bien definidas a través de árboles en una interfaz muy intuitiva.

Ahora, sin embargo, los bots que más predominan son aquellos que cuentan con un motor de IA para interpretar los mensajes entrantes. Esto permite tener bots más inteligentes, flexibles y amigables para el usuario final. Debido a estos aspectos, se decide evaluar más en detalle estas plataformas con aprendizaje automático para crear chatbots. Podemos ver este enfoque en el siguiente apartado.

2.1.1 Herramientas de desarrollo de chatbots con IA

En este apartado se describen las herramientas más usadas de desarrollo de chatbots con motores de IA para el procesamiento del lenguaje natural.

Para entender mejor cómo funcionan estas herramientas, a continuación se describen dos de los conceptos más importantes con los que trabajan:

1. *Intents (intenciones):* señalan las intenciones del usuario, es decir, lo que éste quiere expresar con sus mensajes. Cada intent se define a partir de varias frases de entrenamiento, especificadas previamente, que se usarán para entrenar un modelo de IA el cual permitirá buscar emparejamientos entre los mensajes del usuario y los intents definidos.

   Por ejemplo: El mensaje "Quiero alquilar un coche" se asociará al intent "Alquilar".

2. *Entities (entidades):* son los tipos de los distintos parámetros que contienen los intents. Se usan para extraer palabras relevantes de las frases que los

usuarios envían al bot. En cada entidad se especifican un conjunto de palabras y sinónimos de un ámbito concreto.

Por ejemplo: Del mensaje "Quiero alquilar un coche" podremos obtener qué vehículo quiere alquilar el usuario ("coche") definiendo una entidad "vehículos".

A continuación, se exponen las herramientas actuales más representativas para construir chatbots.

Dialogflow

Dialogflow [12] es un framework para la creación de chatbots de Google. En esta plataforma los bots se organizan en agentes y en cada agente se definirá un intent por cada intención del usuario que se pretenda capturar, además, se definirán las entidades para extraer la información que se precise de los mensajes del usuario.

Con todo esto, Dialogflow creará un modelo de IA con el cual se capturan las intenciones y entidades de los mensajes de los usuarios que se comuniquen con el bot. Una vez se determina la intención del usuario, Dialogflow enviará una respuesta determinada asociada. Esta respuesta bien puede obtenerse tras predefinirlas en cada intent o tras realizar una llamada a un servicio externo.

También, pueden establecerse contextos basados en recordar intenciones y parámetros previos del usuario para guiar el flujo de la conversación. Además, esta herramienta permite conectar los chatbots con multitud de redes sociales y aplicaciones de mensajería instantánea.

Por otra parte, la conexión con un servicio externo hace posible que el bot aumente las funcionalidades que puede ofrecer a los usuarios. Por ejemplo, si conectamos el bot a una base de datos que contiene reservas de vuelos se podrá recordar la fecha en la que un determinado usuario reservó un vuelo.

De esta forma, con Dialogflow se puede crear de forma fácil, rápida y sin conocimientos en IA un bot que puede interactuar con los usuarios manteniendo una conversación estructurada, flexible y con sentido (contexto).

Lex

LEX [13] es el framework que ha desarrollado Amazon para crear bots. Funciona de la misma manera que Dialogflow haciendo uso de intents y entities.

Por otra parte, en cuanto a las diferencias con Dialogflow, podemos encontrar que el manejo de contextos, aquí, viene determinado por la recopilación de atributos de aplicación o sesión y, además, la conexión con servicios externos no se realiza de forma directa sino a través de las llamadas funciones Lambda [14], un servicio de AWS [15] para ejecutar código en respuesta a eventos.

Watson

Al igual que los dos anteriores, Watson [16] también es una herramienta para crear chatbots, en este caso creada por IBM. Su funcionamiento sigue la misma mecánica descrita en las herramientas anteriores.

En cuanto a establecer conexiones con servicios externos, es muy parecido a Lex. Sin embargo, la creación de contextos es más sofisticada que este, estableciéndose mediante árboles de decisión.

LUIS

LUIS [17] es un servicio de Microsoft basado en aprendizaje automático para el procesamiento del lenguaje natural. Su funcionamiento se basa en los principios anteriores descritos: intents y entities.

Sin embargo, podemos ver que LUIS es diferente del resto de las herramientas anteriores debido a que no maneja contextos ni respuestas, sólo podrá ser usado como servicio externo para interpretar la intención del usuario y obtener información relevante de los mensajes del usuario a través de entities; es decir, no tendrá ningún mecanismo para controlar el flujo de las conversaciones ni de las respuestas.

Tabla comparativa

A continuación, se muestra en la tabla 1 una comparativa de todos las herramientas de desarrollo de chatbots analizados en esta sección.

En primer lugar, se indica el tipo concreto de herramienta (framework, plataforma o servicio), después se identifican las herramientas que hacen uso de patrones y/o del procesamiento del lenguaje natural. Por otra parte se señalan las herramientas que, además de texto, reconocen voz. Además se muestran las que tienen soporte para intents/entities y respuestas. Finalmente, se indica el número de lenguajes e integraciones con redes sociales o aplicaciones de mesnajería que soportan.

	Dialogflow	Watson	Lex	LUIS	FlowXO	Landbot	Chatfuel	SmartLoop	Botkit (parte de Microsoft Bot Framework)
Tipo (Framework, Plataforma, Servicio)	P	P	P	S	P	P	P	P	F
Expresiones regulares/patrones		✓		✓	✓	✓		✓	✓
NLP	✓	✓	✓	✓			✓	✓	
Reconocimiento de voz	✓	✓	✓	✓					
Soporte para intents	✓	✓	✓	✓	✓			✓	
Soporte para entities	✓	✓	✓	✓	✓	✓		✓	
Especificación de respuestas del bot	✓	✓	✓		✓	✓	✓	✓	
Número de lenguajes: alto (≥10), medio (<10)	a	m		a	a		a		
Integración con redes sociales/aplicaciones de mensajería/websites: alta (≥10), media (<10), 1 (representado con logo)	a	m	m		m			m	m

Tabla 1: Comparación herramientas de creación de chatbots

2.1.2 Conclusiones

En este caso, con todas las tecnologías anteriores descritas y teniendo en cuenta los objetivos del proyecto, se escoge realizar bots que puedan interpretar el lenguaje natural a través de un motor de IA para que resulten lo más flexibles y amigables para el usuario final.

Por otro lado, centrándonos en las tecnologías que usan IA, la creación de un bot complejo con LUIS presenta más dificultad que cualquiera de las otras opciones debido a toda la lógica que deberá implementarse para proporcionar respuestas al usuario y manejar contextos. Por otra parte, el manejo de contextos con Lex también es pobre si lo comparamos con Watson o Dialogflow.

Así que, finalmente, nos encontramos con Watson o Dialogflow, frameworks que quedan muy igualados en cuanto a facilidad en el manejo de los contextos de las conversaciones. Sin embargo, Dialogflow será finalmente el framework elegido para la creación de los bots ya que las conexiones con servicios externos se implementan mucho más fácilmente que el resto y cuenta con un mayor número de redes sociales y aplicaciones de mensajería con las que puede integrarse.

2.2 Plataformas de mensajería

Las plataformas de mensajería son un medio de comunicación en tiempo real para el envío de mensajes de texto, imágenes, videos o documentos [18] entre varios usuarios.

Estas aplicaciones son comúnmente usadas a diario por millones de usuarios y, debido a que uno de los objetivos de este proyecto es el de proporcionar al usuario un entorno conocido y fácil de usar para realizar consultas a una fuente de datos, se propone el uso de estas aplicaciones como intermediarias entre el usuario y nuestro sistema.

A continuación se describirán varias de las plataformas de mensajería más conocidas.

Whatsapp

Whatsapp [19] es la aplicación de mensajería para smartphones con mayor número de usuarios. Actualmente es gratuita [20].

A parte de los mensajes textuales, en esta aplicación se pueden enviar otros elementos tales como stickers, imágenes, vídeos o localizaciones, además de poder realizar llamadas de voz y vídeo por IP [21]. Los datos se almacenan localmente por lo que las copias de seguridad son necesarias para preservar los distintos chats.

También, puede ser usada desde el PC, sin embargo, es necesario que el móvil que contenga la cuenta de whatsapp del usuario esté conectado a Internet en todo momento [21].

Telegram

Telegram es una aplicación de mensajería instantánea multiplataforma de código abierto y completamente gratuita [21]. Posee las mismas funcionalidades de Whatsapp, sin embargo, extiende y mejora algunas como no ser necesario conectar el móvil del usuario a internet para chatear desde el ordenador o no necesitar copias de seguridad, ya que todos los mensajes quedan alojados en la nube [21].

Al igual que Whatsapp, también cuenta con la posibilidad de crear grupos con multitud de personas, sin embargo, Telegram cuenta con más opciones y mecanismos de administración de grupos. A su vez, existen los canales, chats a

los que un usuario puede unirse pero sólo el propietario del canal podrá enviar mensajes, suelen usarse como canales para publicar noticias.

Por otro lado, se ha convertido en una de las plataformas más populares a la hora de comunicarse con bots debido a su amplio y documentado soporte para este tipo de servicios [21].

Facebook Messenger

Facebook [22] es el nombre de la red social de la empresa con el mismo nombre. Esta red social conecta personas a través de relaciones de amistad y permite compartir con éstas información, noticias o contenidos multimedia [23].

Por otro lado, Facebook cuenta con una plataforma de mensajería instantánea, Facebook Messenger [24], que permite comunicar de manera privada a los usuarios de Facebook, aunque cuenta con muchas menos funcionalidades que Whatsapp y Telegram.

Por otra parte, sí cuenta con soporte para la integración de bots. Esto es porque que numerosas empresas usan Facebook como escaparate para vender sus productos y así, con la inclusión de bots, éstas pueden gestionar de forma más eficiente la atención al usuario.

2.2.1 Conclusiones

Debido al gran potencial que tienen estas aplicaciones en cuanto a usabilidad y popularidad, se decide vincular este proyecto a estas plataformas con el fin de que los usuarios puedan hacer uso de servicios externos a través de un entorno conocido.

Whatsapp es la aplicación de mensajería que tiene mayor número de usuarios, sin embargo, no tiene soporte para desarrollar y vincular bots como Telegram o Facebook Messenger.

Por otro lado, aunque Facebook Messenger cuenta con soporte para bots, el ofrecido por Telegram es mucho mayor y cuidado. Además, Telegram brinda mejor experiencia de usuario a la hora de chatear ya que es la aplicación de mensajería con mayor número de características y funcionalidades (que aumentan de manera exponencial con cada actualización).

Por tanto, la aplicación elegida con mejor ratio en cuanto a número de usuarios, funcionalidades y soporte para bots es Telegram.

2.3 Herramientas de consulta en lenguaje natural

A continuación, se describen otros proyectos que, como el descrito en este documento, usan el lenguaje natural para obtener información de una fuente de datos.

2.3.1 SQLizer

SQLizer [25] es una herramienta para sintetizar consultas SQL a partir de lenguaje natural. Para el análisis sintáctico usan SEMPRE [26] junto con modelos pre-entrenados de la librería CoreNLP [27]. Así, esta herramienta es independiente de la base de datos que se quiera consultar y por tanto, no requiere ningún entrenamiento previo.

Este proyecto sigue un proceso iterativo que crea un esqueleto SQL con 'huecos' que se irán rellenando con entradas basadas en el propio esquema inicial y en la base de datos asociada y, finalmente, se verifica si es necesario 'reparar' la consulta y empezar con el proceso descrito de nuevo [25].

2.3.2 Seq2SQL

Seq2SQL [28] es una herramienta que usa una red neuronal para interpretar el lenguaje natural y obtener una consulta SQL.

El enfoque utiliza WikiSQL [28], creado por los mismos autores, el cual es una colección de fuentes múltiples de preguntas en lenguaje natural con sus correspondientes consultas y tablas SQL.

Divide las consultas SQL en tres partes distintas (operaciones de agregación, columnas *SELECT*, es decir, los datos que se quieren seleccionar y cláusulas *WHERE,* los filtros que se desean añadir a la consulta) que serán rellenadas por la red neuronal con los distintos elementos que las constituyen.

2.3.3 nl2sql

nl2sql [29] es una aplicación que ofrece una interfaz para realizar consultas en lenguaje natural a tablas de la base de datos WikiSQL, convirtiéndolas en sentencias SQL. Se basada en el algoritmo Coarse2Fine de Inteligencia Artificial para procesar las consultas entrantes y generara la sentencia SQL resultante.

2.3.4 Conclusiones

Los proyectos anteriormente citados tienen como objetivo crear un motor de traducción de lenguaje natural a SQL desde cero, usando Inteligencia Artificial, sin poner el foco en usuarios sin conocimientos técnicos. Sin embargo, el proyecto que describe este documento no pretende el desarrollo de un motor de IA propio sino usar frameworks como Dialogflow y centrarse, por tanto, en acercar esta traducción (lenguaje natural a lenguaje específico de consulta) a cualquier tipo de usuario.

Además, se pretende que esta herramienta se use con datos abiertos y extender las funcionalidades de consulta de estos proyectos aportando nuevas formas de visualización de información como son gráficas y mapas.

2.4 Resumen y conclusiones

La creación de un chatbot con un motor de inteligencia artificial bien construido, potente y flexible es una ardua tarea. Sin embargo, esta tarea se vuelve muy sencilla y visual gracias a los frameworks que proporcionan compañías como Google.

Por otro lado, la sociedad está en continuo contacto con las aplicaciones de mensajería, en las cuales se hace uso del lenguaje natural para comunicarse con otras personas. En cuanto a Telegram, gracias al número de usuarios familiarizados esta aplicación, el extenso soporte para bots y las distintas opciones de interacción como los botones, encontramos que es una herramienta muy potente como mediadora entre usuarios y servicios.

Por tanto, para la realización de este proyecto se elige usar la herramienta de creación de chatbots de Google, Dialogflow y la aplicación de mensajería Telegram.

Capítulo 3
Enfoque

En este capítulo podemos encontrar una breve descripción de cómo funciona el sistema en general y qué partes lo componen (sección 3.1). Además, se muestra el funcionamiento concreto de GENBOT y los chatbots que genera éste (secciones 3.2 y 3.3 respectivamente).

3.1 Visión general

Como se ha explicado en la sección 1.2, este proyecto busca acercar la consulta y visualización de datos abiertos a cualquier tipo de usuario. Esto se llevará a cabo a través de un chatbot integrado con Telegram, sin embargo, será necesario adaptar éste al conjunto de datos al que se quiere consultar. Con este fin nace GENBOT, un "super" bot que, dado un conjunto de datos en formato CSV y otros parámetros adicionales (descritos en el apartado 3), crea automáticamente un chatbot específico de consulta y visualización.

Cabe destacar que existen dos tipos de roles diferentes que interaccionan con el sistema. En primer lugar, habrá un usuario que interacciona con GENBOT, interesado en crear chatbots de consulta específicos y tiene un mínimo de conocimientos técnicos en la consulta de datos. Por último, está el usuario final para el que se crea el chatbot específico, el cual no es necesario que tenga conocimientos técnicos de ninguna clase.

A continuación, se detalla de forma breve el funcionamiento global del sistema y las partes que lo componen:

1. En primer lugar, un usuario interesado en crear un bot de consulta y visualización a una fuente de datos abriría una conversación a través de Telegram con GENBOT.

2. El segundo paso es especificar toda la información que necesita GENBOT para funcionar:

a. Desde el chat de GENBOT: El conjunto de datos que se quiere consultar junto con una serie de información adicional para crear la base de datos derivada y generar mapas (ver apartado 3.2.1).
b. Desde el archivo de configuración llamado *properties:* los tipos de consulta que el chatbot resultante reconocerá (ver apartado 3.2.1).

3. Después, GENBOT creará automáticamente un chatbot para el conjunto de datos especificado.

4. Finalmente, cualquier usuario podrá iniciar una conversación en Telegram con el chatbot creado por GENBOT y comenzar a consultar y visualizar el conjunto de datos especificado en el primer paso. (Se necesita vinculación manual previa con Telegram y subida a Dialogflow, ver limitaciones apartado 6.2)

Podemos ver los pasos anteriores ilustrados en la figura 1.

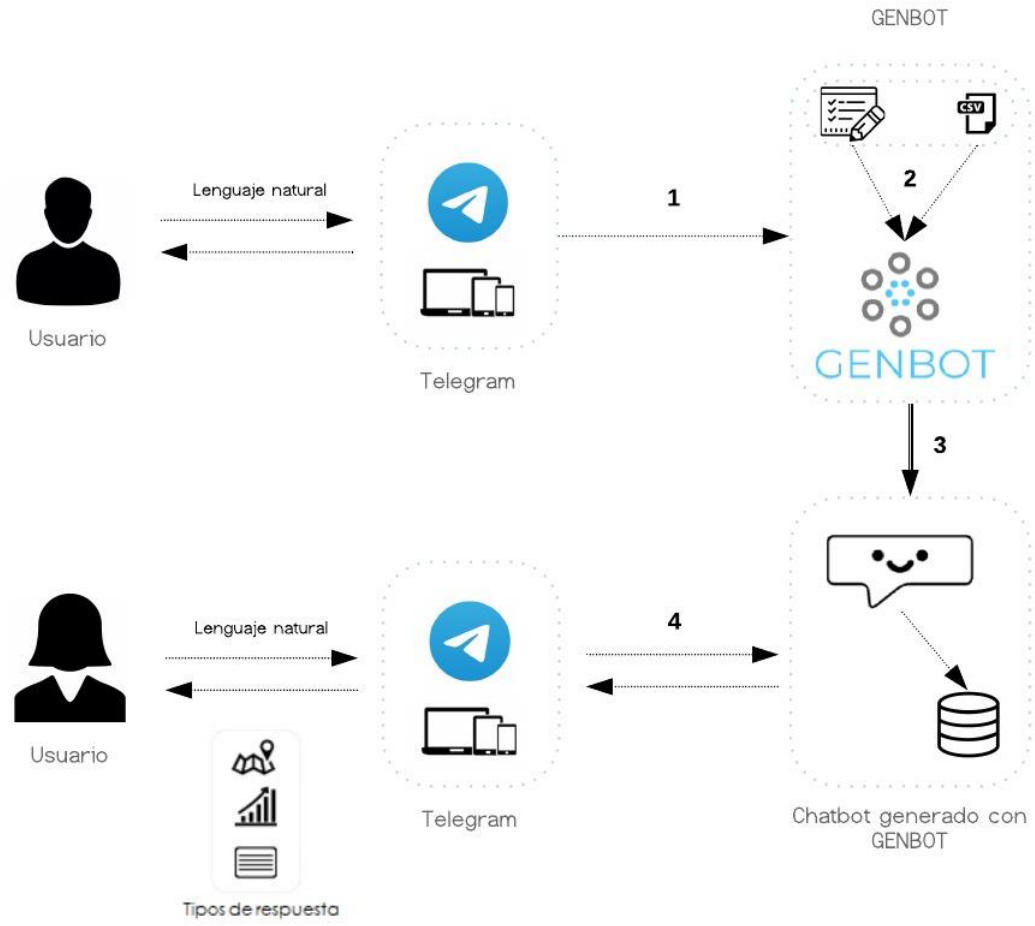


Figura 1: Funcionamiento global del sistema

3.2 GENBOT

GENBOT es el encargado de crear de manera automatizada cualquier chatbot asociado a cualquier fuente de datos en formato CSV.

Como podemos ver en la figura 2, el funcionamiento de GENBOT es el siguiente:

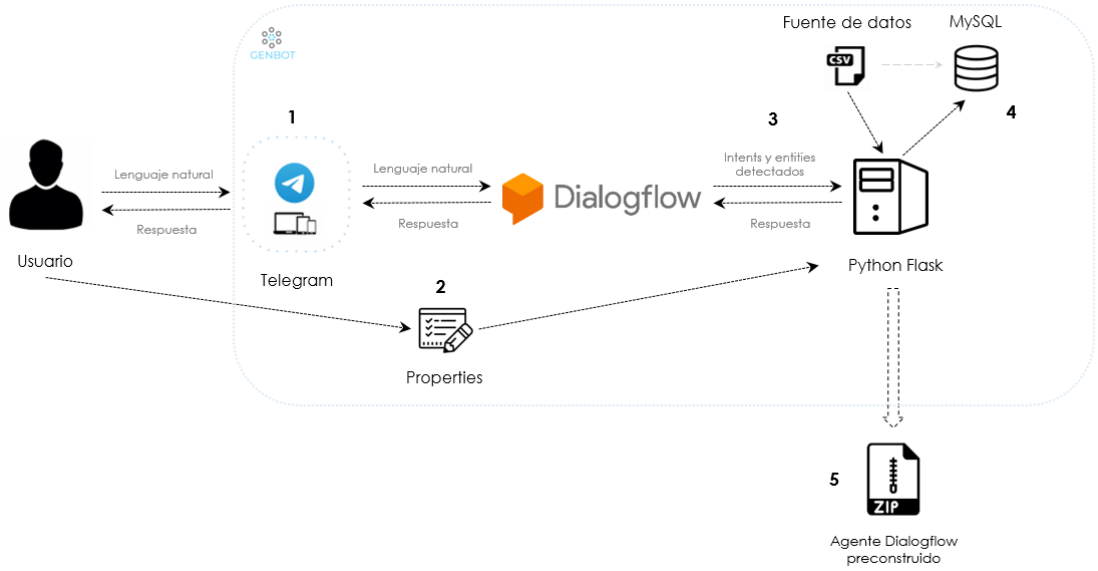


Figura 2: Detalle funcionamiento GENBOT

1. Un usuario interesado inicia una conversación con GENBOT a través del bot *@super_genbot* de Telegram.

2. Además, el usuario interesado edita el archivo *properties* para proporcionar información a GENBOT sobre cómo serán las consultas a realizar a la fuente de datos (ver apartado 3.2.1).

3. El usuario, a través de Telegram, habla con GENBOT. Estos mensajes llegarán a Dialogflow, el cual los interpretará y enviará esta información al servidor. Después, en el lado del servidor, se analizarán estas interpretaciones y se generan las respuestas adecuadas para enviar al usuario. De esta manera, el usuario habla con GENBOT y proporciona toda la información necesaria para generar un chatbot específico (ver sección 3).

4. Una vez el usuario interesado ha proporcionado a GENBOT toda la información necesaria, es decir, ha concluido la conversación con éxito (ver sección 3), se genera una base de datos a partir de la fuente de datos que se haya proporcionado en el paso anterior. Esta base de datos será utilizada después por el chatbot autogenerado resultante para realizar consultas SQL.

5. Finalmente, el servidor en Python genera un agente preconstruido de Dialogflow.

3.2.1 Datos solicitados por GENBOT

GENBOT requerirá distintos datos al usuario para dar como resultado un agente de Dialogflow preconstruido. Por un lado, solicitará datos a través del chat en Telegram y, por otro, estudiará el contenido de un archivo *properties.*

Chat de Telegram

Los datos que solicita a través del chat con Telegram son los siguientes:

1. Nombre de la base de datos que se creará en el servidor de GENBOT para realizar consultas SQL a través del lenguaje natural.

2. Nombre de la tabla donde se guardará la información del CSV proporcionado.

3. URL del archivo CSV el cual es la fuente de datos que el usuario quiere consultar y visualizar.

4. Nombres de las columnas de la tabla si el CSV no los proporciona.

5. Sinónimos para llamar a las distintas columnas de las que se tienen datos en el CSV. Estas palabras serán necesarias para identificar las distintas formas que tiene el usuario de hacer referencia a una columna.

6. El tipo de las columnas de la tabla de la base de datos, en el caso de que los proporcionados por GENBOT no sean los adecuados. GENBOT procesará cada columna, inferirá un tipo automáticamente y se lo mostrará al usuario, si el usuario no está de acuerdo con el tipo podrá proponer otro.

7. El tipo de entidades de Dialogflow que usará el agente para reconocer los datos que porpocione el usuario en el chatbot, en el caso de que los proporcionados por GENBOT no sean los adecuados (véase apartado 4.1.5). Al igual que en el punto anterior, GENBOT inferirá los tipos de entidades predefinidas de Dialogflow más adecuados de manera automática, pero el usuario final podrá cambiarlos.

8. Las columnas que contengan coordenadas en el caso de querer generar mapas.

9. La columna que contiene el valor por el que el usuario pedirá visualizar un elemento en un mapa.

   Por ejemplo: En el mensaje "Muéstrame en un mapa la farmacia de Antonio Pérez", "Antonio Pérez" es un valor de la columna *titular_farmacia.*

10. Las columnas del elemento a visualizar en mapas de las que se quiere obtener información adicional. Esta información aparecerá en las ventanas emergentes que pueden desplegarse tras pulsar el icono de posición en los mapas dinámicos que se generen.

    Por ejemplo: Con el mensaje "Muéstrame en un mapa la farmacia de Antonio Pérez", el usuario quiere visualizar dónde se encuentra la farmacia elegida. Pero, además, quiere que se le muestre una ventana emergente que contenga información sobre la calle y el número de la farmacia. Por tanto, en este paso, deberá proporcionar al bot las columnas *calle* y *número.*

Archivo properties

En el archivo de configuración *properties,* el usuario introducirá toda la información referente a las distintas consultas que manejará el chatbot resultante de GENBOT.

En lugar de proporcionar esta información a través del chat de Telegram, se ha visto más adecuado tratar con un archivo de texto. Esto se debe a que la información que hay que incluir puede ser bastante extensa y complicada de escribir desde un chat, además, incluir saltos de línea desde el chat de Telegram con algunos dispositivos puede ser una tarea complicada de configurar.

Las figuras 4, 5 y 6 ilustran los distintos tipos de consultas que se pueden realizar en los chatbots. En cada figura, se muestra en primer lugar el texto tipo que proporcionará el usuario al chatbot a través de Telegram. A la hora de hablar con el chatbot, no es necesario que se introduzca el mismo texto de forma exacta ya que el motor de Dialogflow es bastante flexible a la hora de analizar las sentencias enviadas por el usuario. Después, se tiene el texto que será necesario añadir en el archivo *properties,* el cual tiene una estructura que se define en párrafos siguientes y, por último, se encuentra la sentencia SQL asociada al texto, que será ejecutada por el servidor.

El primer tipo de consultas (figura 4) es el más sencillo, se caracterizan porque no contienen ningún filtro, es decir, no se especifican valores concretos en la consulta (una sentencia SQL sin *where*). En este caso, a modo de ejemplo, un usuario quiere conocer todos los nombres (consulta en figura 4) de la tabla de la figura 3.

Table: users
name
username
favorite_color
favorite_fruit

Figura 3: Tabla users

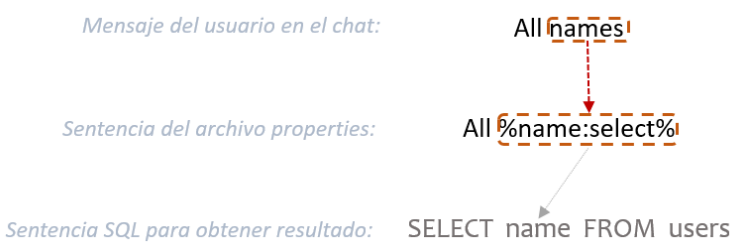


Figura 4: Estructura consulta para obtener todos los valores de un campo

El segundo tipo (figura 5), se caracteriza por contener valores concretos, es decir, filtros. Ahora, siguiendo con el ejemplo anterior, el usuario pide sólo los nombres de personas a las que les gusta el color azul y los plátanos.

Figura 5: Estructura consulta simple con filtros

Por último, el tercer tipo (figura 6) es muy parecido al segundo, también contiene valores específicos por los que filtrar. Sin embargo, en este caso, se especifica además el nombre del campo del valor por el que filtrar. Es decir, el valor 'Sandra99' se corresponde con el campo *username,* el cual no hay que confundir con el campo *name*.

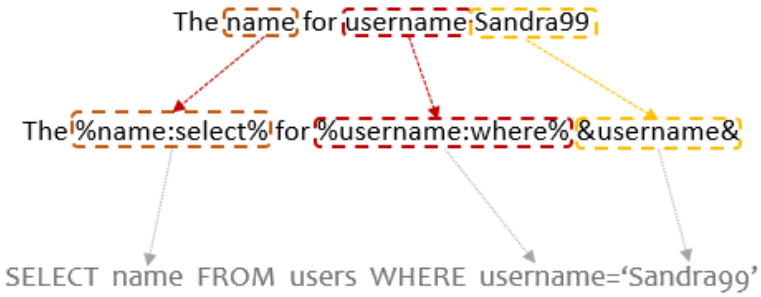


Figura 6: Estructura consulta compleja con filtros

Por tanto, el usuario creador del chatbot, a la hora de introducir una frase en el archivo *properties,* debe tener en cuenta tres aspectos: (1) los campos de los que se desea obtener información, (2) los valores por los que filtrar y (3) los campos asociados a los filtros.

De esta manera, el formato asociado a los aspectos anteriores es el siguiente:

1. %nombre_columna:select%
2. &nombre_columna&
3. %nombre_columna:select%

Siguiendo este formato específico, el usuario creador de chatbots debe introducir las distintas frases que serán reconocidas por este bot. En el apartado 4.1.5 se muestra en detalle cómo el sistema trata estas frases.

3.2.2 Flujo de diálogo con GENBOT

El diálogo que mantiene el usuario con GENBOT se ciñe exclusivamente para proporcionar información acerca de la fuente de datos que se quiere consultar. El flujo de diálogo con este chatbot está predefinido y se puede visualizar en la figura 7.

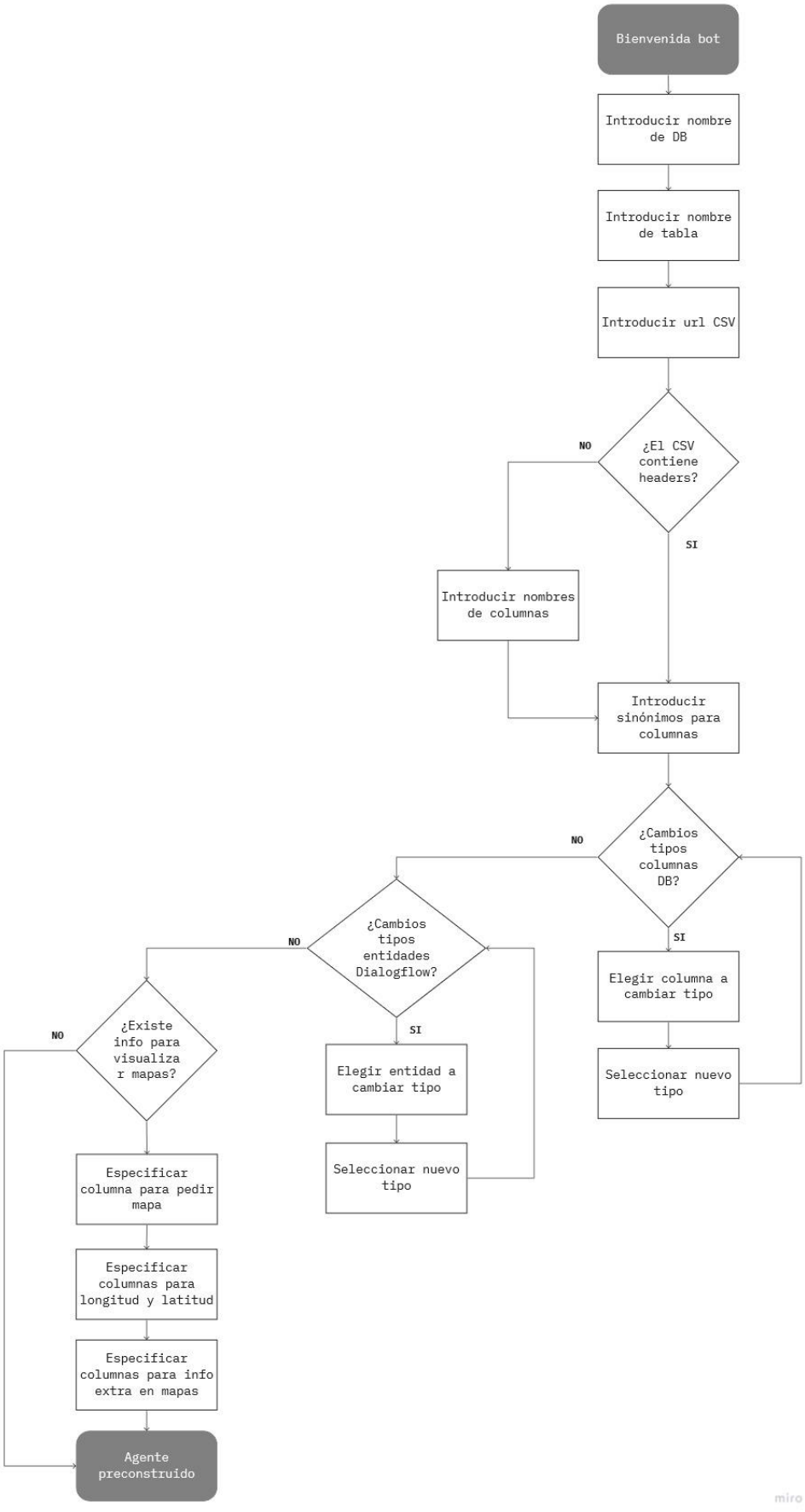


Figura 7: Flujo de diálogo de GENBOT

3.3 Chatbots autogenerados

Después de acabar la conversación con GENBOT, el usuario obtendrá un agente de dialogflow preconfigurado asociado a una fuente de datos listo subirlo a Dialogflow y vincularlo a Telegram. Una vez hecho esto, el usuario podrá chatear con él para consultar y visualizar información de la fuente de datos asociada.

Figura 8: Detalle funcionamiento bot generado por GENBOT

3.3.1 Funcionalidades

A continuación, se detallan las distintas acciones que estos chatbots pueden realizar, ilustradas en la imagen 8.

Consultas

Los datos que generen las consultas podrán visualizarse a través de un mensaje de texto en el chat de Telegram (figura 9). Si éstos son muy extensos se proporcionarán en un fichero de texto que se enviará a la conversación (figura 10).

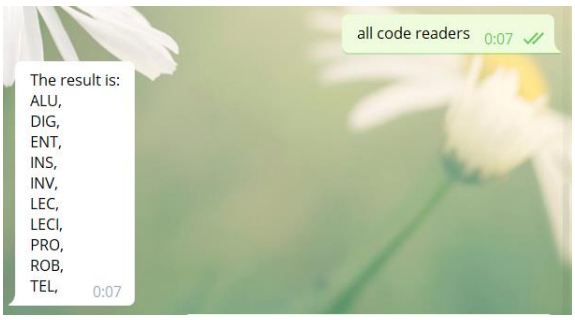


Figura 9: Consulta “obtener todos los códigos de lector” con resultado en chat

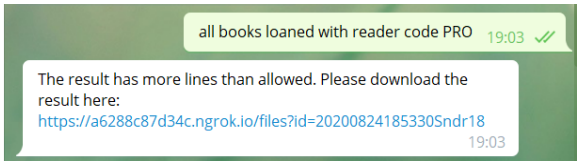


Figura 10: Consulta “obtener todos los libros para un código de lector dato” con resultado en fichero de texto

Mapas

El chatbot puede proporcionar información visual sobre coordenadas, éstas se muestran de manera visual a través de un mapa. Esta funcionalidad queda sujeta a la existencia de coordenadas en el dataset y a su correcta identificación a través de GENBOT.

Las figuras 11 y 12 muestran los distintos tipos de mapas que ofrecen estos chatbots.

Figura 11: Mapa estático creado con Mapbox

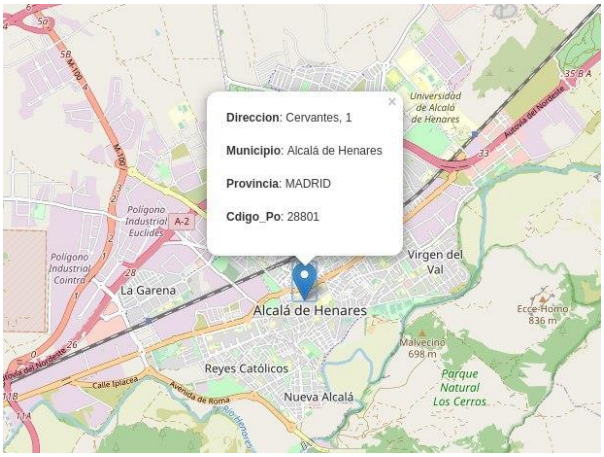
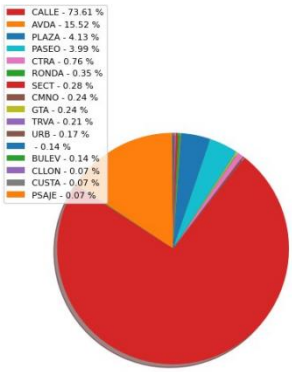
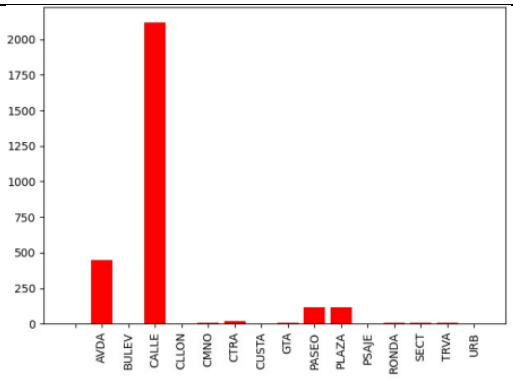


Figura 12: Mapa interactivo creado con Folium

Gráficos

El usuario puede pedir distintos tipos de gráficos para una o dos variables.

Gráficos de una variable
 Gráfico de tarta
 Histograma

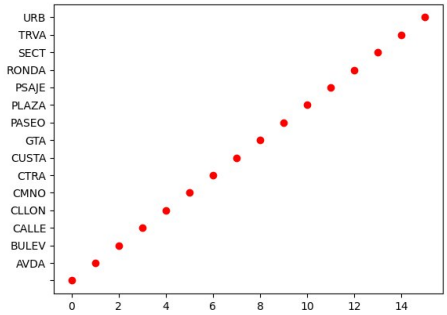


Gráfico de puntos (series)

El eje x identifica cada uno de los registros de la base de datos y el eje y muestra el valor asociado para la columna indicada al bot. Se usan puntos.

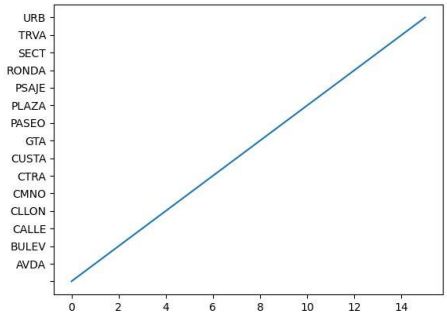


Gráfico de líneas (series)

El eje x identifica cada uno de los registros de la base de datos y el eje y muestra el valor asociado para la columna indicada al bot. Se usan líneas.

Tabla 2: Gráficos de una variable

Gráficos de dos variables

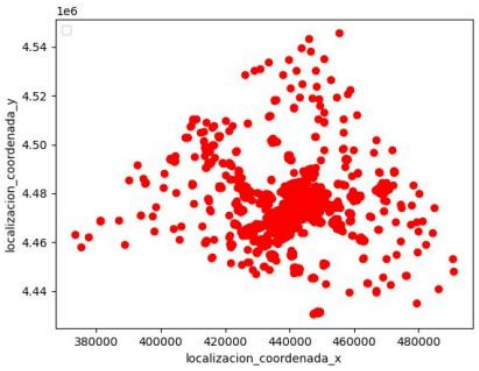



Gráfico de dispersión (X vs Y)

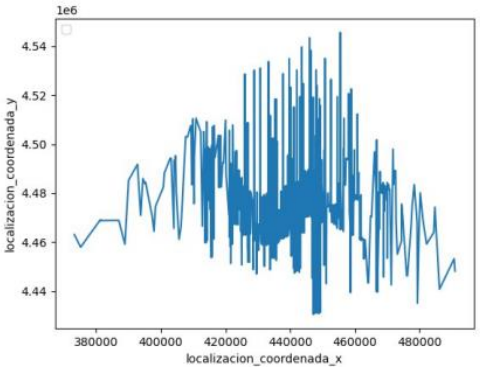


Gráfico de dispersión con líneas de conexión (X vs Y)

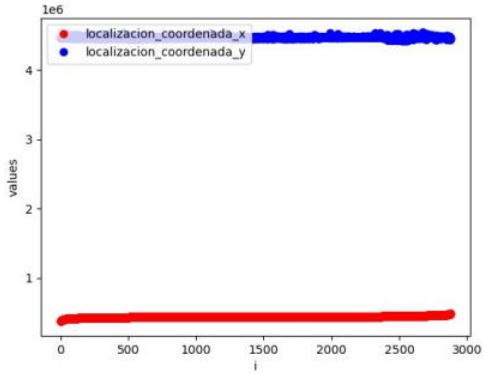


Gráfico de puntos (series)

El eje *x* identifica cada uno de los registros de la base de datos y el eje *y* muestra el valor asociado para las dos columnas indicadas en el chat. Cada columna se identifica con un color distinto. Se usan puntos.

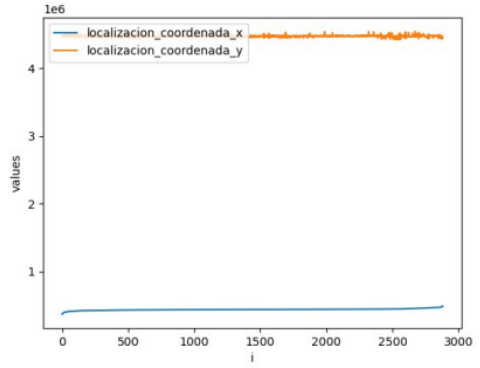


Gráfico de línea (series)

El eje *x* identifica cada uno de los registros de la base de datos y el eje *y* muestra el valor asociado para las dos columnas indicadas en el chat. Cada columna se identifica con un color distinto. Se usan líneas.

Tabla 3: Gráficos de dos variables

Ayuda

Por último, el usuario puede pedir más información o ayuda al chatbot. La figura 13 muestra el formato de la ayuda que se proporciona. En primer lugar, se

proporciona información sobre la fuente de datos asociada, después se da información sobre cómo pedir por gráficos y mapas (si corresponde) y cómo realizar consultas.

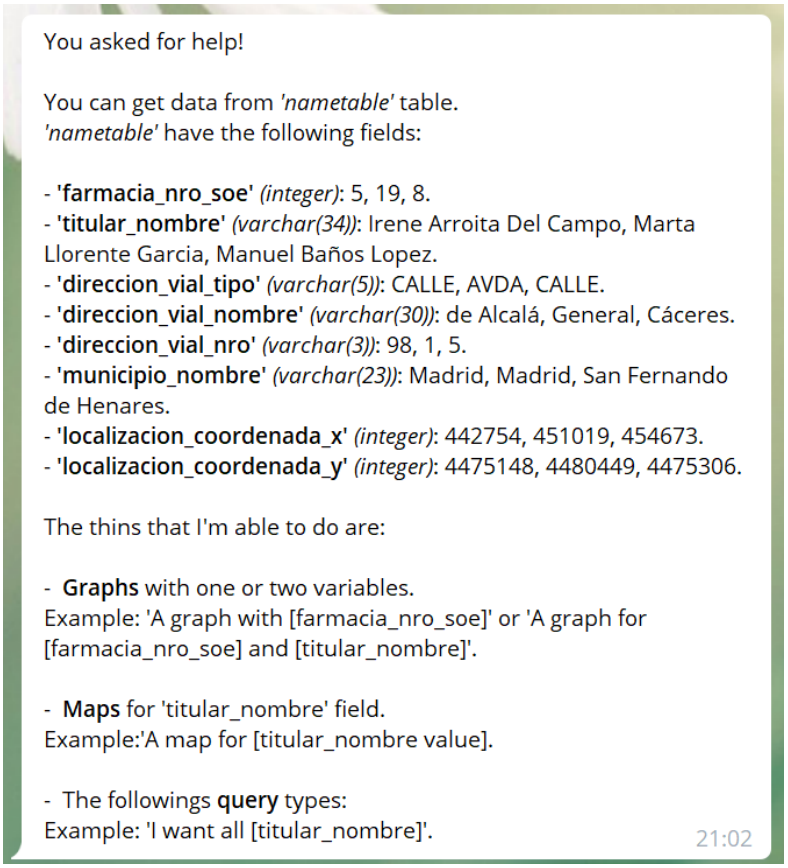


Figura 13: Ayuda en chatbots autogenerados

3.3.2 Flujo de diálogo chatbots específicos

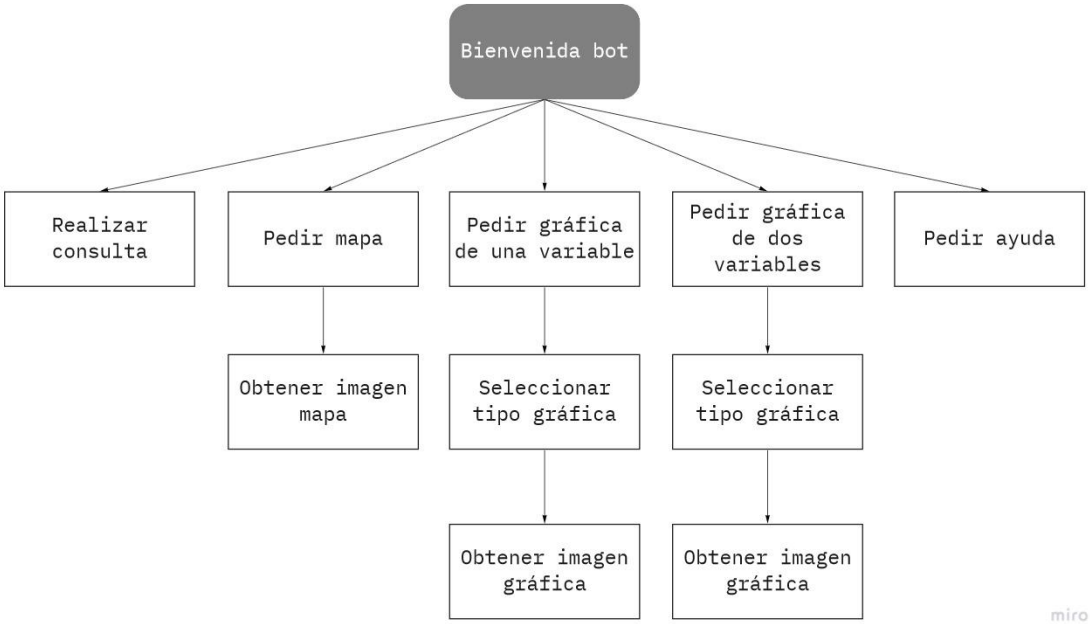


Figura 14: Flujo de diálogo de chatbots autogenerados

El diálogo que mantendrá el usuario con estos chatbots tendrá como fin hacer uso de las funcionalidades de consulta y visualización de la información de la fuente de datos asociada (figura 14).

El flujo del diálogo en estos chatbots no está predeterminado como en GENBOT. En cualquier caso, el usuario puede acceder a cualquier funcionalidad.

Capítulo 4
Arquitectura y herramienta

En este capítulo se describen la arquitectura (Sección 4.1) y la herramienta (Sección 4.2) tanto de GENBOT como de los chatbots autogenerados.

4.1 Arquitectura

En primer lugar, se ha separado la lógica de GENBOT de la de los chatbots autogenerados. Para ello, se han creado dos servidores distintos en Python utilizando Flask [30]. Uno para recibir los mensajes del usuario cuando se comunique con GENBOT y otro para los mensajes de todos los chatbots autogenerados que se creen.

Ambos tipos de chatbots siguen una arquitectura de Webhook. Esta arquitectura crea un sistema de notificaciones mediante HTTP con la que una parte del sistema espera notificaciones de la otra.

Por una parte, está Dialogflow, encargado de procesar los mensajes en lenguaje natural del usuario (envío de notificaciones), y por otra, se encuentran los servidores en Python (recibo de notificaciones). Con esta arquitectura, los servidores en Python esperan por notifiaciones de Dialogflow.

En concreto, cuando Dialogflow recibe un mensaje del usuario, éste lo interpreta y envía al servidor en Python esta información a través de una petición POST. Después, el servidor estudia lo que se le ha hecho llegar y envía a Dialogflow una respuesta que después será remitida al usuario.

4.1.1 Modelo de Dialogflow en chatbots autogenerados

Cada funcionalidad del chatbot (intención de usuario) se corresponde con un intent de Dialogflow distinto. Las distintas intenciones, y por tanto, los intents definidos en Dialogflow son: obtener datos en texto sobre una consulta, obtener mapa, obtener gráfico de una variable, obtener gráfico de dos variables, bienvenida, ayuda y obtener recurso.

Todos los chatbots autogenerados se comunican con el mismo servidor. Esto implica que los modelos de Dialogflow para cada chatbot deban personalizarse. Esto es necesario para que el servidor identifique la base de datos y tabla a la que dirigir las consultas SQL resultantes. Así, serán los intents de cada chatbot los encargados de contar con toda la información necesaria para realizar estas consultas a la base de datos.

En este caso, cada intent asociado a una intención de usuario que implique lanzar una consulta SQL (consultas en texto plano, gráficos y mapas) contendrá como parámetros fijos el nombre de la tabla y el de la base de datos a los que dirigir la sentencia SQL.

En los siguientes apartados de este capítulo se explica con más detalle de qué se componen los intents y las sentencias SQL realizadas de las distintas funcionalidades de los chatbots.

4.1.2 Base de datos

En cuanto a los datos consultados en los chatbots autogenerados, se ha elegido almacenar éstos en una base de datos relacional en Mysql. Aunque el proceso de consulta lo lleva a cabo el servidor de los chatbots autogenerados, la tarea de creación de la base de datos es realizada por el servidor de GENBOT.

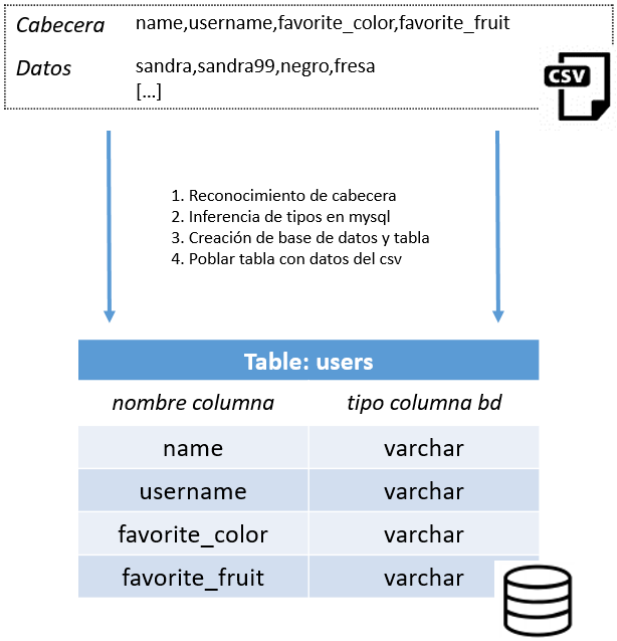


Figura 15: Proceso de creación de base de datos a partir de CSV

Para crear esta base de datos, el servidor de GENBOT usa la información que el usuario le proporciona en el chat: nombre de la base de datos, nombre de la tabla y nombres de las columnas. Por otra parte, para poblar de información la tabla creada, el servidor exportará a ésta los datos del CSV que el usuario previamente le ha proporcionado. Todo ello se realiza a través de una sentencia específica de Mysql para este trabajo. Este procedimiento puede visualizarse en la figura 15.

Después, una vez creado el chatbot autogenerado y, en consecuencia, la base de datos en Mysql, el proceso que se realiza cuando el usuario interactúa con el chatbot para la consulta y visualización de información es el siguiente (los pasos pueden verse ilustrados en la figura 8):

1. El usuario envía un mensaje a través de Telegram.
2. Dialogflow interpreta el mensaje, lo fragmenta y envía esta información al servidor
3. El servidor analiza la información de Dialogflow y crea una consulta en SQL que ejecuta contra la base de datos en Mysql. Este paso se realiza de igual manera para gráficos, mapas o datos en texto plano.
4. La consulta SQL es ejecutada y los resultados que devuelve la base de datos se envían de vuelta al usuario (a través de Dialogflow).

4.1.3 Gráficos

Para la creación de los gráficos de los chatbots autogenerados se usa una librería de Python llamada Matplotlib [31]. Los gráficos pueden ser de una o dos variables y los distintos tipos se exponen en el apartado 3.3.1.

Debido a los timeouts fijados por Dialogflow y con imposibilidad de cambio, ha sido necesario rediseñar la parte del envío de imágenes de los gráficos creados con Matplotlib. A menudo, las operaciones que realizaba Matplotlib para la generación del gráfico excedían los timeouts de respuesta de Dialogflow. Debido a esto, los gráficos nunca llegaban al usuario final.

Para solventar este problema, en lugar de enviar la imagen en sí, se envía un botón de Telegram con un id de identificación único para usuario y recurso. Tras pulsar el usuario este botón, se hace una petición al servidor para obtener la imagen del gráfico que se corresponda con el id del botón. Si el gráfico aún está procesándose, se le indicará al usuario que espere unos momentos para preguntar de nuevo por él, si por el contrario ya está listo, se le enviará.

Para poder generar los gráficos, ya sea para visualizar una variable o dos, es necesario que el usuario introduzca el nombre de la variable a representar (ver figura 32).

En la figura 16 se puede observar el intent creado para identificar la intención "visualizar un gráfico de dos variables".

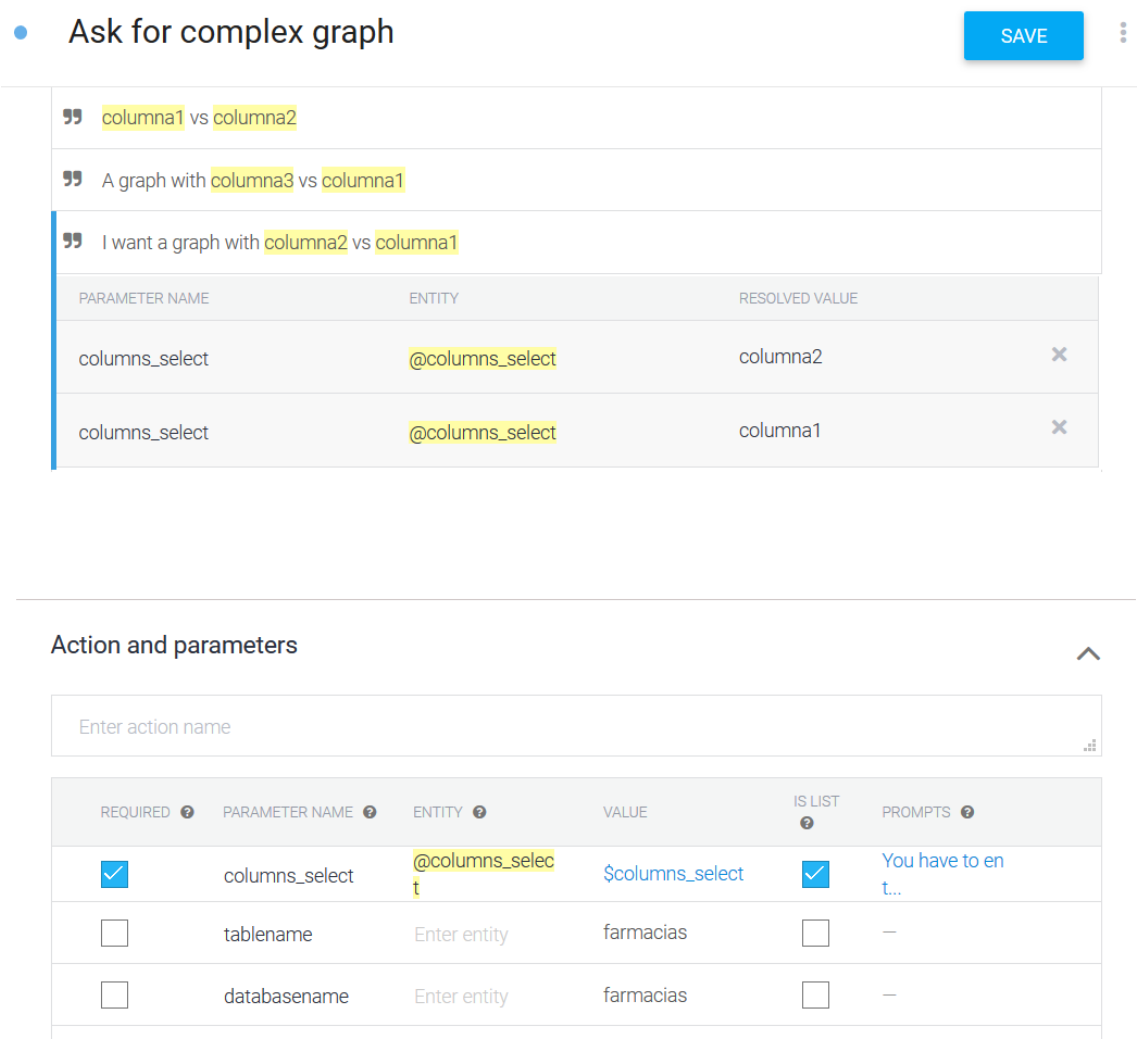


Figura 16: Intent "Ask for complex graph"

En la primera parte de la imagen, están las frases de entrenamiento, éstas se modifican y se combinan para poder incluir todos los campos de los que está formada la fuente de datos, de tal manera que Dialogflow reconozca todas las columnas de la tabla.

En la segunda parte, se encuentran los parámetros. En ellos se guarda el nombre de la base de datos y de la tabla a la que realizar consultas. Además, existe otro parámetro, *columns_select*, que guarda en una lista los campos que quiere visualizar el usuario. Este parámetro tiene asociada la entidad *@columns_select*

la cual está formada por todos las columnas de la tabla y los sinónimos que introdujo el usuario al hablar con GENBOT.

Con toda esta información extraída de los parámetros, la sentencia SQL que se genera es la correspondiente a la figura 17. Cabe destacar que esta consulta es para los gráficos de dos variables, en el caso de los de una variable, sólo habría un campo en la sentencia *SELECT*.

```
USE databasename;
SELECT field1, field2 FROM tablename;
```

Figura 17: Sentencia SQL para crear gráficos

4.1.4 Mapas

Los mapas que genera el chatbot son de dos tipos. En primer lugar se envía una imagen estática que contiene un mapa de la localización pedida por el usuario, donde ésta se identifica con un marcador de posición. Por otro lado, se envía una url, que si se accede a ella desde un navegador, se puede visualizar un mapa interactivo en el que el usuario puede hacer zoom y, además, ver información extra de la localización indicada a través de un popup. Se podrá visualizar hasta un máximo de 10 ubicaciones distintas en el mismo mapa.

Para la creación de mapas estáticos se ha usado la API de Mapbox [32], una plataforma de mapas de cógido abierto. Por otro lado, para crear los mapas interactivos se ha usado Folium [33], una librería de Python que usa Leaflet.js [34] para la creación de mapas. En este caso, Folium genera un archivo html con código javascript usando la librería leaflet.

Aunque al servir mapas no se han tenido problemas de timeouts, se ha decidido usar la técnica descrita en el punto anterior (envío de botón para solicitar recurso), para evitar futuros problemas con las imágenes.

El intent de Dialogflow destinado a reconocer la intención del usuario de obtener un mapa es “Ask for map”, ilustrado en las figuras 18 y 19. En la primera, podemos ver las frases de entrenamiento y en la segunda los parámetros del intent.

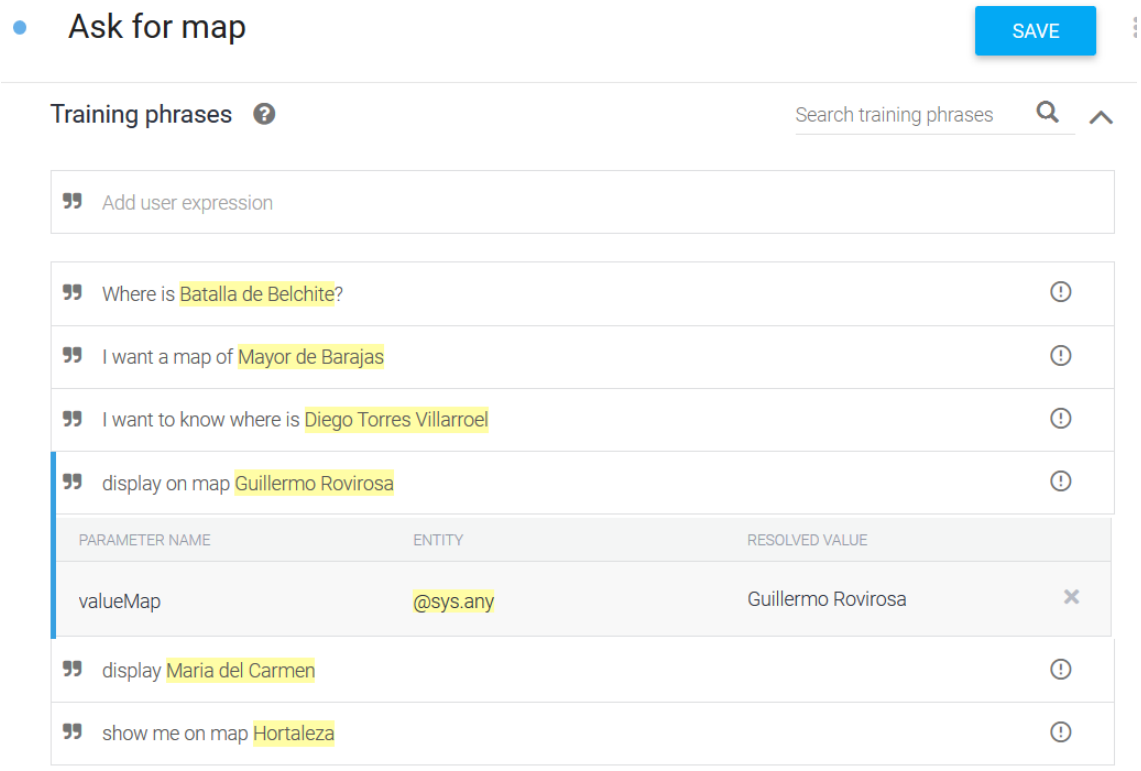


Figura 18: Detalle frases de entrenamiento del intent "Ask for map"

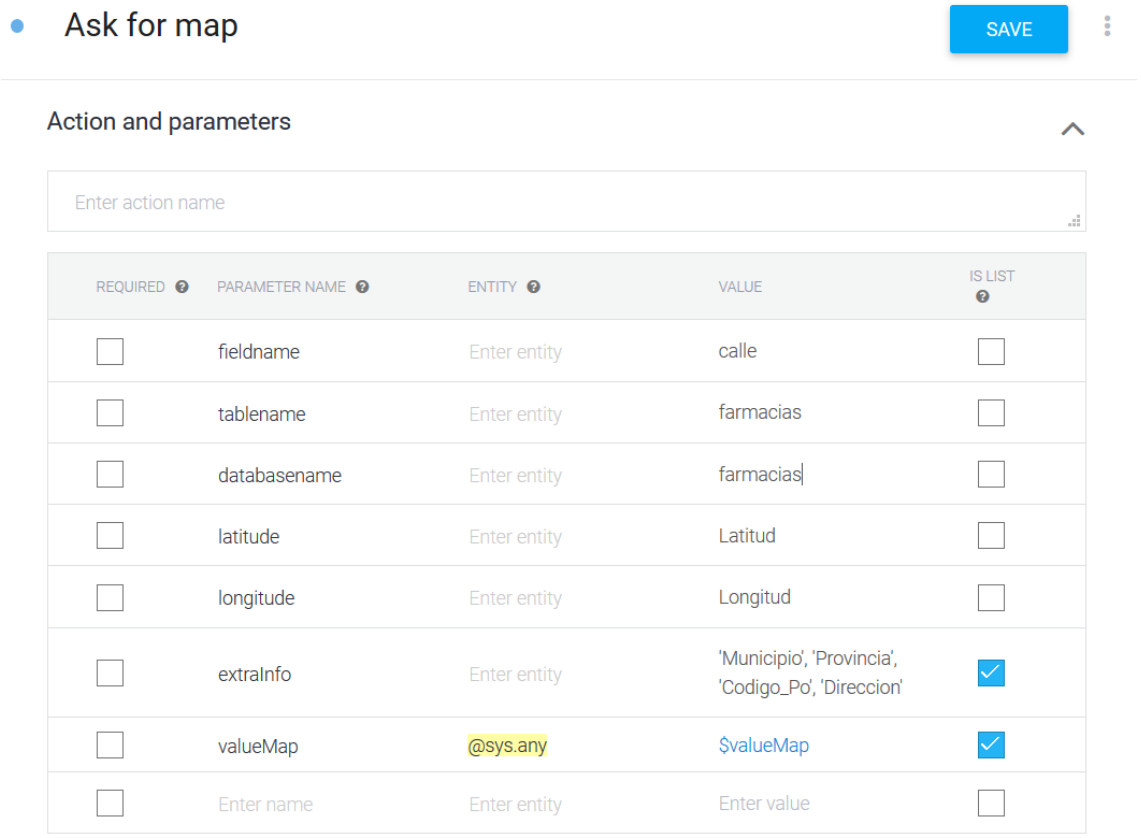


Figura 19: Detalle parámetros del intent "Ask for map"

```
USE db;
SELECT lat, lon, info1, info2 [...] FROM tablename WHERE field='value'
```

Figura 20: Sentencia SQL para crear mapas

A continuación, se muestra en una tabla toda la información que se necesita para la generación de mapas. En ella, se muestra cómo se identifica la información en Dialogflow (figuras 18 y 19) y de la sentencia SQL (figura 20).

Nombre parámetro en Dialogflow	Sentencia SQL	Tipo de dato	Descripción
latitude	*lat*	Nombre de columna	Nombre de las columnas de la tabla donde están todos los valores correspondientes a la coordenada de latitud.
longitude	*lon*	Nombre de columna	Nombre de las columnas de la tabla donde están todos los valores correspondientes a la coordenada de longitud.
extraInfo	*info1, info2* [...]	Nombre de columna	Nombres de las columnas de la tabla de las que se quiere extraer información adicional para rellenar los popups o los bocadillos que acompañan al punto de posición en los mapas interactivos.
fieldname	*field*	Nombre de columna	Nombre de la columna a la que hacen referencia los valores de *valueMap.*
valueMap	*value*	Valor	Valor de la columna especificada en *fieldname.* Se corresponde con el valor que introduce un usuario para pedir por un mapa. Por ejemplo, "Muéstrame en un mapa la farmacia de la calle Liberación". En este caso *Liberación* será un valor de la columna calle. En Dialogflow, se asocia al tipo de entidad *@sys.any* puesto que este valor puede contener cualquier tipo de información. Además, las frases de entrenamiento se crean con ejemplos aleatorios pero reales de estos valores, sacados de la propia fuente de datos.
databasename	*db*	Nombre base de datos	Nombre de la base de datos a la que realizar la consulta.
tablename	*tablename*	Nombre de tabla	Nombre de la tabla a la que realizar la consulta.

Tabla 4: Información encesaria para la generación de mapas

4.1.5 Consultas en texto plano

Para que el usuario pueda realizar consultas a la fuente de datos, es necesario que el creador del chatbot primero complete el archivo *properties* con los distintos tipos de frases que se quiera que el chatbot reconozca como consultas (siguiendo el formato especificado en el apartado 3.2.1).

En primer lugar, el sistema analizará la fuente de datos proporcionada (véase figura 3 a modo de ejemplo) y asociará a cada campo un tipo de entidad de Dialogflow de manera automática. Además, si el usuario considera oportuno, estos tipos de entidad pueden cambiarse durante la conversación con GENBOT.

Dialogflow cuenta con entidades predefinidas y estas serán asignadas a aquellos campos de la fuente de datos con los que los tipos predefinidos coincidan. Por ejemplo, para el campo *favorite_ color*, Dialogflow tiene la entidad *sys.color* (véase figura 21). Sin embargo, otros campos no podrán asociarse a entidades predefinidas y el sistema deberá crear unas personalizadas. Estas entidades se crean tomando valores reales de la propia fuente de datos. Como ejemplo, encontramos los campos *favourite_* fruit y *username* para los cuales no había una entidad predefinida (véase figura 21).

Por otro lado, para reconocer en los mensajes del usuario los campos de la fuente de datos, es necesario crear una entidad que recoja los nombres de éstos y sus sinónimos, proporcionados por el usuario al hablar con GENBOT. En la figura 21 puede verse como ejemplo la entidad *columns.*

Una vez el sistema crea y asocia las entidades correspondientes a cada campo, se da lugar el procesamiento del archivo *properties.* Cada frase es analizada y se crea un intent distinto para reconocer cada una de ellas. Después, para completar el intent es necesario crear distintos parámetros y frases de entrenamiento.

Las frases de entrenamiento creadas dependen de cada una de las palabras especiales de cada consulta, es decir, de cada palabra asociada a un parámetro del intent. El sistema modificará estas palabras en función los valores de la base de datos y creará un conjunto de frases de entrenamiento. En concreto, se generará una frase de entrenamiento por cada combinación posible de valores.

Para la creación de estas frases diferenciamos dos tipos de entidades: (1) la entidad asociada a nombres de columnas de la fuente de datos y (2) las entidades asociadas a los valores de estas columnas. Al crear las frases de entrenamiento, las palabras asociadas al primer tipo de entidad se modifican en función de los distintos sinónimos definidos para cada campo. Por otra parte, las palabras

asociadas al segundo tipo de entidades se modifican a partir de los distintos valores que puede tomar cada entidad, estos valores se extraen directamente de ejemplos reales de la fuente de datos.

Por último, existe una excepción a la hora de crear los intents: se ligarán al mismo intent aquellas frases que contengan los mismos tipos de entidades.

Podemos ver un ejemplo ilustrativo en las figuras 21 y 22.

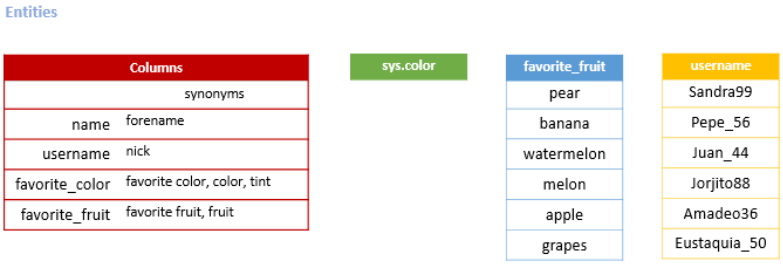


Figura 21: Entidades. Caso práctico tabla users

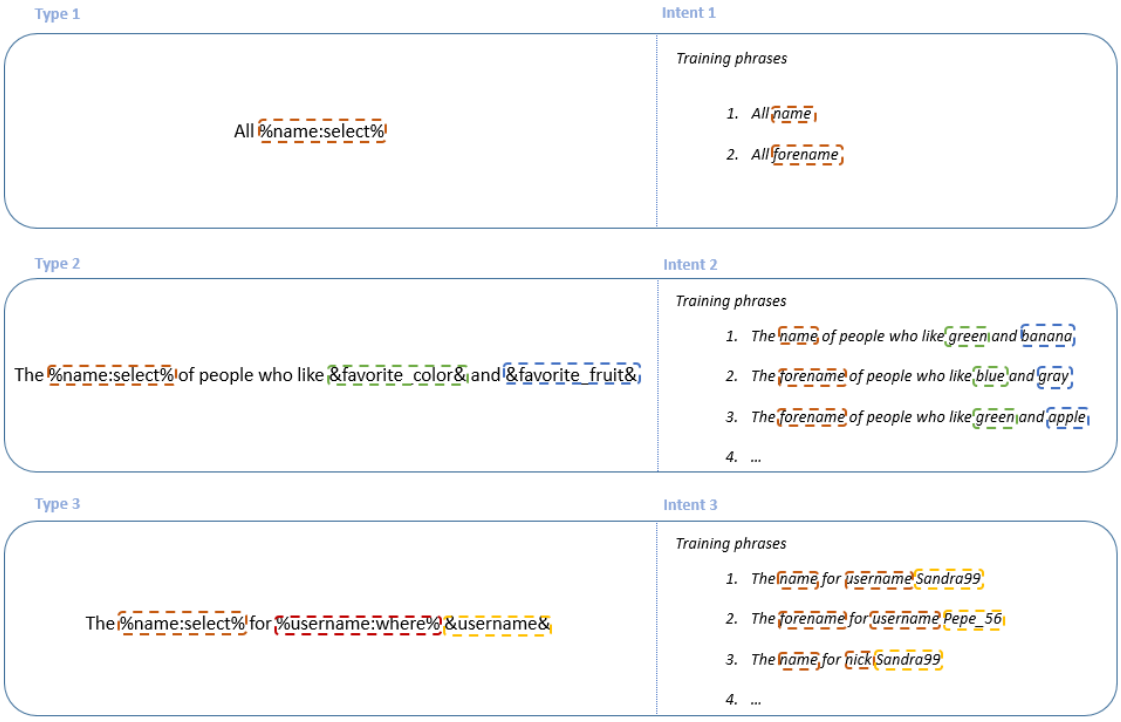


Figura 22: Intents y frases de entrenamiento. Caso práctico tabla users

Después de crear las frases de entrenamiento, es necesario definir todos los parámetros que contendrá cada intent. A estos parámeros se les asocia un tipo de entidad y servirán para recoger los datos necesarios del mensaje del usuario para, posteriormente, crear una sentencia en SQL. Finalmente, esta sentencia SQL se ejecuta contra la base de datos creada a partir del CSV proporcionado por el usuario. A continuación, en la tabla 5 se ve cómo se relaciona la información

anteriormente descrita, es decir, los distintos tipos de parámetros de cada intent, el tipo de entidad asociada y cómo se relacionan con el archivo *properties* y con la sentencia SQL (figura 23).

Las columnas asociadas a los valores de filtros no se asocian a ningún parámetro, solamente le sirven al sistema para identificar donde hay nombres de columnas y generar frases de entrenamiento más variadas.

Formato palabra archivo *properties*	Nombre del parámetro en Dialogflow	Tipo de entidad	Sentencia SQL
%nombre_columna:select%	columns_select	*columns*	*columns_select*
%nombre_columna:where%	-	-	*columns_where*
&nombre_columna&	Nombre de la columna a la que se asocia el parámetro	*Entidad predefinida o personalizada*	Valor de la columna a la que se asocia el parámetro
-	*databasename*	-	*db*
-	*tablename*	-	*tablename*

Tabla 5: Relación entre los parámetros de los intents y la sentencia SQL

```
USE db;
SELECT columns_select[1], columns_select[2] [...] FROM tablename
WHERE columns_where[1]='value1' and columns_where[2]='value2' [...]
```

Figura 23: Sentencia SQL para consultas en texto plano

Las imágenes con las consultas SQL se han simplificado para una mayor comprensión, pero a la hora de ejecutarlas en el servidor se incluyen mecanismos para solventar ciertos errores del usuario a la hora de escribir por chat. En concreto, se obvian las mayúsculas y minúsculas y, además, los valores que introduce el usuario y los de la base de datos se evalúan en cuanto a un valor de similitud, por medio de la función SOUNDEX de mysql. De esta manera, las consultas se vuelven más flexibles a los posibles errores de escritura de los usuarios en los chats.

Finalmente, los resultados obtenidos tras ejecutar las sentencias SQL son enviados al usuario a través de texto plano en un mensaje de Telegram (figura 9). Sin embargo, si se obtienen más de 20 resultados distintos, el sistema enviará una url en lugar de un mensaje en texto plano. A través de la url proporcionada, el usuario puede descargar un fichero de texto con todos los resultados de la consulta realizada (figura 10).

4.1.6 Creación del archivo comprimido del chatbot autogenerado

Una vez se ha explicado de qué elementos estará formado el chatbot autogenerado, es preciso detallar el proceso para crear éstos. Cabe destacar que, debido a las limitaciones de Dialogflow, es imposible crear agentes de forma automática de principio a fin.

Para este propósito, los servidores de GENBOT se conectan con un agente de Dialogflow propio, previamente creado, a través de la API del framework de Google. Con esto se crearán los distintos intents y entidades explicados en apartados anteriores.

Finalmente, una vez creados todos los elementos, se importa el agente de Dialogflow resultante en formato .zip y se envía al usuario final.

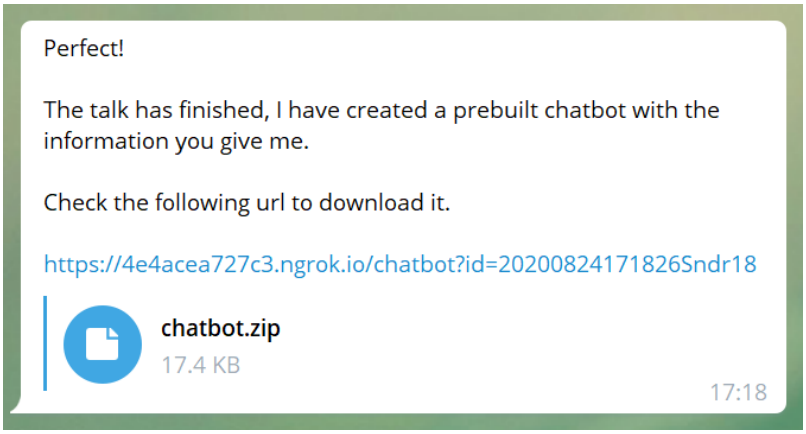


Figura 24: Mensaje de GENBOT para descargar el chatbot autogenerado resultante

4.2 Herramienta

A continuación, se explica el front-end del sistema, es decir, cómo interactúan los usuarios con GENBOT y los chatbots que genera éste. En ambos casos, el usuario interactuará a través de la aplicación de mensajería Telegram manteniendo una conversación en inglés con el usuario.

4.2.1 GENBOT

El usuario habla con GENBOT para proporcionarle toda la información necesaria para crear un chatbot asociado a una fuente de datos. A continuación, se ilustran algunos ejemplos de interacción con el usuario.

En la figura 25 se puede ver como el usuario comienza la interacción con GENBOT. Después, en la figura 26 se ilustra la manera en la que el usuario introduce los sinónimos asociados a un campo concreto de la fuente de datos.

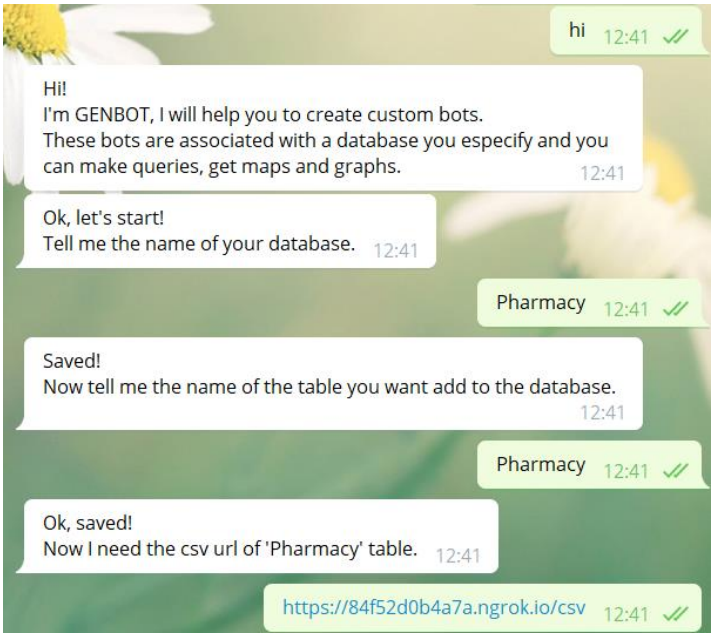


Figura 25: Comienzo conversación con GENBOT

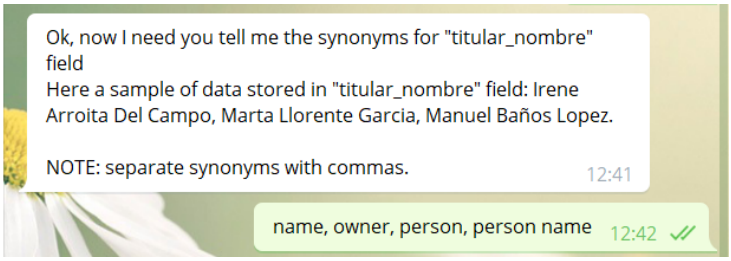


Figura 26: Interacción con GENBOT para introducir sinónimos para el campo "titular_nombre"

Por otro lado, tras proporcionar la fuente de datos, GENBOT infiere de forma automática los tipos adecuados de las columnas de la base de datos que se creará y de los tipos de entidades asociados a cada campo. Sin embargo, como ya se ha especificado en puntos anteriores, el usuario puede cambiar estos tipos.

En primer lugar, en la figura 27 se ve como el usuario dice a GENBOT que quiere cambiar los tipos de las columnas de la base de datos. Después, GENBOT, a través de botones (*inline buttons)*, le pide al usuario que pulse el campo al que quiere cambiar el tipo. Finalmente, GENBOT le proporciona (con *keyboard buttons)* los distintos tipos que puede elegir.

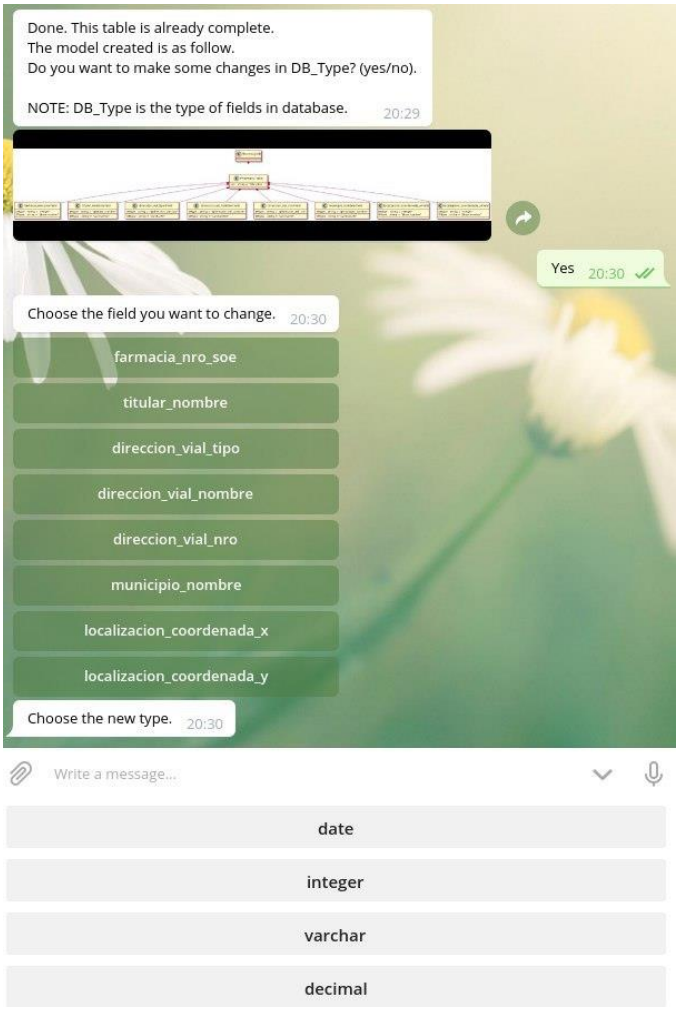


Figura 27: Interacción con GENBOT para el cambio de los tipos de la base de datos

Por otro lado, el cambio de tipos de las entidades de dialogflow asociadas a cada campo se realiza del mismo modo que el descrito en el párrafo anterior.

Finalmente, para la generación de mapas, es necesario especificar los campos que contendrán las coordenadas de las posiciones a marcar. En la figura 28 se muestra este aspecto. El usuario deberá elegir los campos correspondientes a las coordenadas a través de los botones presentados por GENBOT.

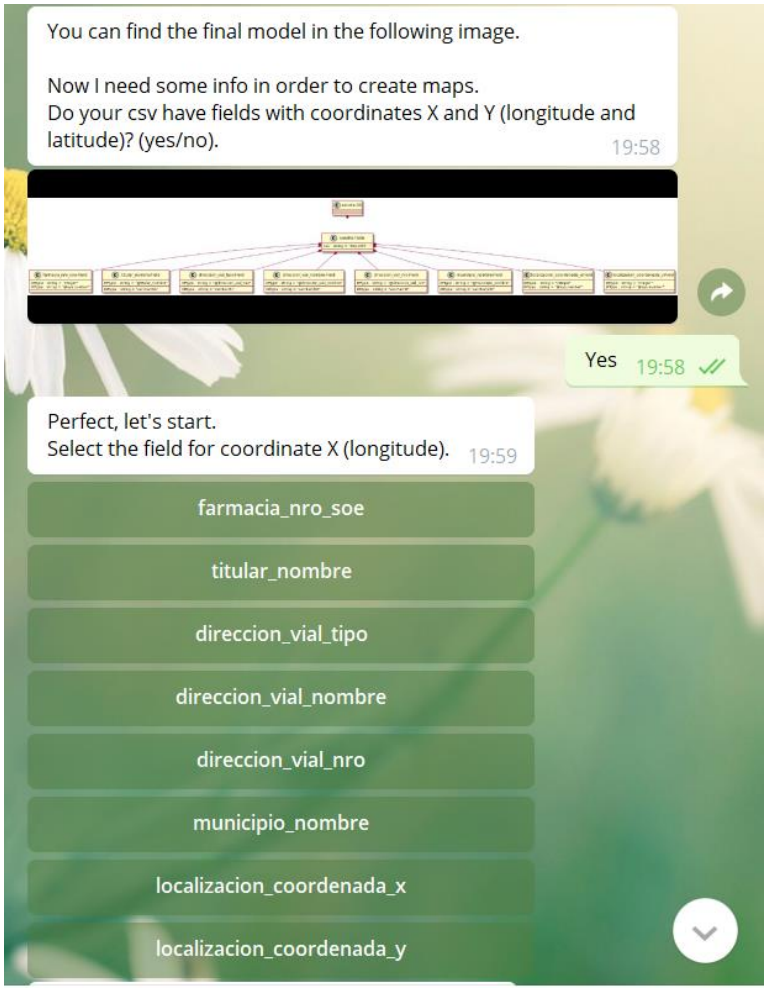


Figura 28: Interacción con GENBOT para selección del campo de coordenadas de longitud para mapas

Además, en la figura 29 el usuario especifica las columnas de las cuales se quiere extraer información para detallar posiciones en mapas. Se puede ver un ejemplo de cómo se mostrará esta información en la figura 12.

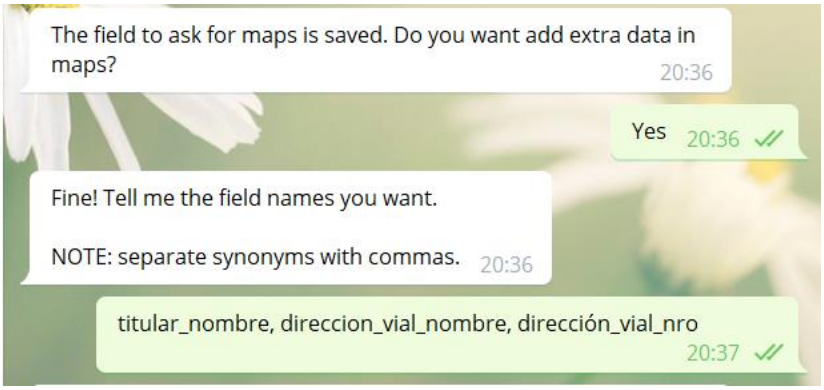


Figura 29: Interacción con GENBOT para proporcionar información adicional en mapas

4.2.2 Chatbots autogenerados

Estos chatbots se asocian a una fuente de datos y permiten al usuario realizar consultas en lenguaje natural y obtener mapas y gráficos.

En primer lugar, en la figura 30 se muestra el inicio de una conversación con estos chatbots, después, en la figura 31 se observa cómo el usuario pide ayuda al bot.

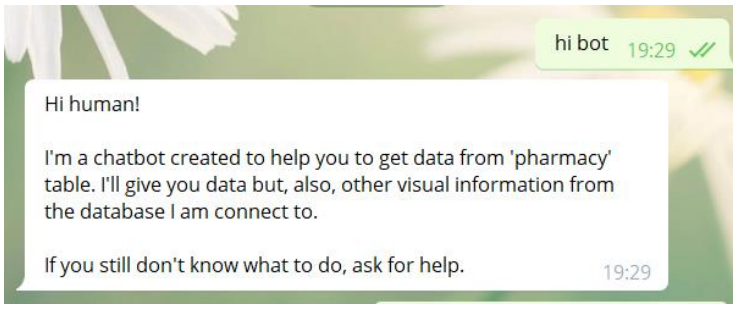


Figura 30: Inicio conversación chatbot autogenerado

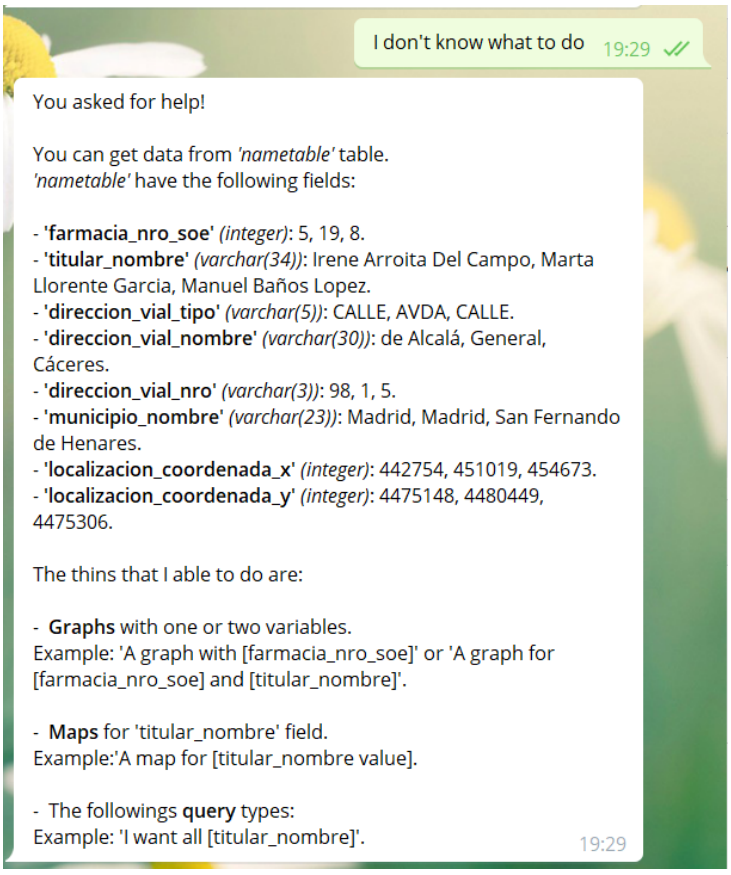


Figura 31: Interacción chatbot autogenerado para pedir ayuda

En cuanto a consultas, se presenta un ejemplo en las figuras 9 y 10. Mientras en la primera imagen los resultados se muestran en el mismo chat, en la segunda se le proporciona al usuario un archivo de texto. Esto es debido a que la segunda

consulta ha excedido el número máximo de registros a mostrar a través del chat. El contenido del archivo se visualiza en la figura 32.

Figura 32: Contenido archivo de resultados

En cuanto a los gráficos, en la figura 33 se muestra la interacción que sigue un usuario para pedir al chatbot un gráfico con una variable. Primero, el usuario pide un gráfico, después, el bot le pregunta (a través de botones) el tipo de gráfico que quiere obtener el usuario. Por último, el bot le proporciona un botón para obtener la imagen del gráfico pedido.

A su vez, en la figura 33 se muestra el mensaje "Wait a while until the server completes the graph". Este mensaje aparecería cuando el usuario pulsa el botón de obtener gráfico y el gráfico aún no ha terminado de generarse.

Por otra parte, en la figura 34 se muestra un ejemplo de cómo un usuario interacciona con el chatbot para pedir un mapa. En primer lugar, el bot muestra un botón para obtener el mapa. Después de pulsarlo (y si está ya listo), el bot servirá el mapa pedido. El mapa estático se envía como imagen, y el dinámico es accesible a través de una url proporcionada.

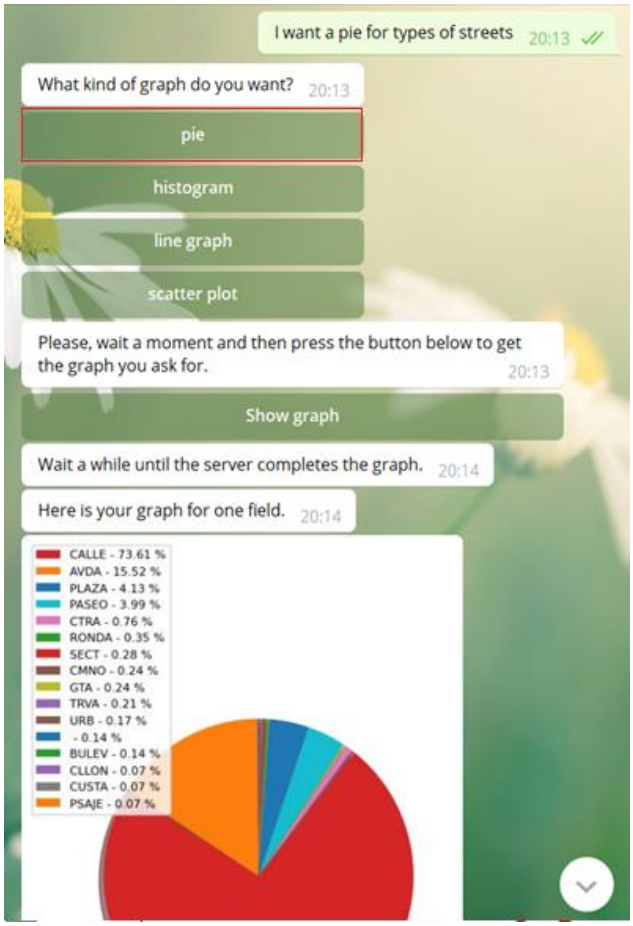


Figura 33: Interacción con chatbot autogenerado para obtener gráficos

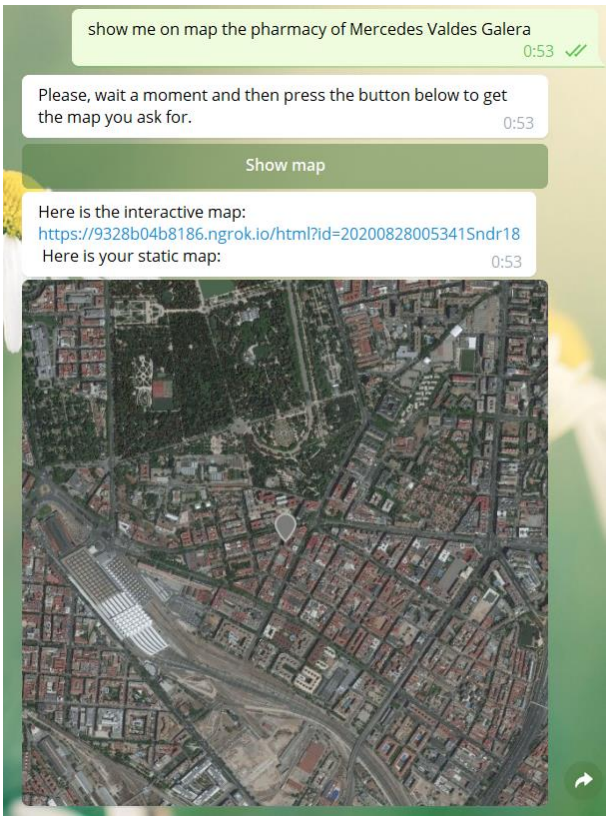


Figura 34: Interacción con chatbot autogenerado para obtener mapas

4.3 Conclusiones

Actualmente, los servidores del proyecto se encuentran alojados en un ordenador personal y su dirección pública cambia cada vez que se levantan. Por tanto, cada vez que se incian (imprescindible para hablar con los chatbots), es necesario cambiar en el proyecto y agentes de Dialogflow ciertos parámetros asociados a la dirección pública del servidor.

Una de las soluciones a esto podría ser alojar el proyecto en servidores con direcciones públicas sin interrupción del servicio. No obstante podría crear complicaciones de presupuesto, almacenamiento y escalabilidad. Por tanto, la solución deseada implica preparar el proyecto para permitir el despliegue automático del servicio (ver trabajos futuros, sección 6.3) para particulares, donde a su vez, éstos puedan modificar y ampliar las funcionalidades según sus gustos y necesidades.

El cógido del proyecto se encuentra en el siguiente repositorio de GitHub: https://github.com/Snd18/GENBOT_public

GENBOT está asociado al siguiente usuario de telegram: @super_genbot

Capítulo 5
Caso de uso y pruebas

En este capítulo se exponen distintos caso de uso para probar el funcionamiento completo de la herramienta. El apartado 5.1 se centra en un caso de uso en el que se realizan consultas en texto plano y se pide visualizar distintos tipos gráficos. Por otro lado, en el apartado 5.2 se muestra un caso de uso específico para visualizar mapas, donde también se muestran cómo se pueden cambiar los tipos inferidos de las columnas y entidades. Para ambos casos, se usarán datos abiertos de la Comunidad de Madrid.

5.1 Caso de uso 1: Mapas y gráficos

Para este caso de uso se ha elegido un dataset abierto de Madrid que contiene la localización de todas las farmacias madrileñas. Con él, se prueba la funcionalidad de visualización de mapas y gráficos que tiene la herramienta.

En primer lugar, hay que modificar el archivo properties. Para este caso, como no queremos hacer ninguna consulta en texto plano, se procede a dejar en blanco su contenido. De esta forma, el chatbot resultante sólo reconocerá intenciones de usuario relacionadas con mapas y gráficos.

A continuación, es necesario hablar con GENBOT. En la conversación se detalla la siguiente información: nombre de la base de datos, nombre de la tabla, url del dataset (figura 25), sinónimos de los campos (figura 26 y tabla 6), el nombre del campo por el que se pedirán los mapas (figura 35), los campos que contienen los valores de latitud y longitud (figura 28) y los campos para mostrar información extra en los mapas (figura 29).

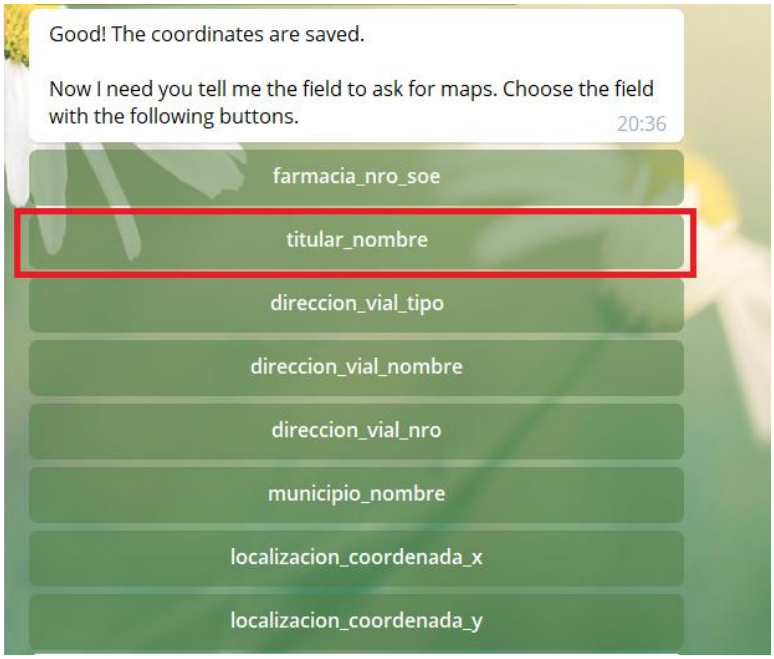


Figura 35: Selección campo por el que se pedirán mapas

Nombre campo	Sinónimos
farmacia_nro_soe	id
titular_nombre	owner, person, person name
direccion_vial_tipo	type of street
direccion_vial_nombre	street, address
direccion_vial_nro	num, number
municipio_nombre	city, town
localización_coordenada_x	x, lon, longitude
localización_coordenada_y	y, lat, latitude

Tabla 6: Sinónimos campos de la tabla *pharmacy*

Se recuerda que no ha sido necesario proporcionar los nombres de las columnas de la tabla puesto que el dataset ya las incluía.

Por otro lado, los tipos de las columnas de la tabla y los tipos de las entidades en Dialogflow inferidos por la herramienta son los correctos. Sin embargo, a modo de prueba, se procede a cambiar el tipo de columna y de entidad del campo *farmacia_nro_soe*.

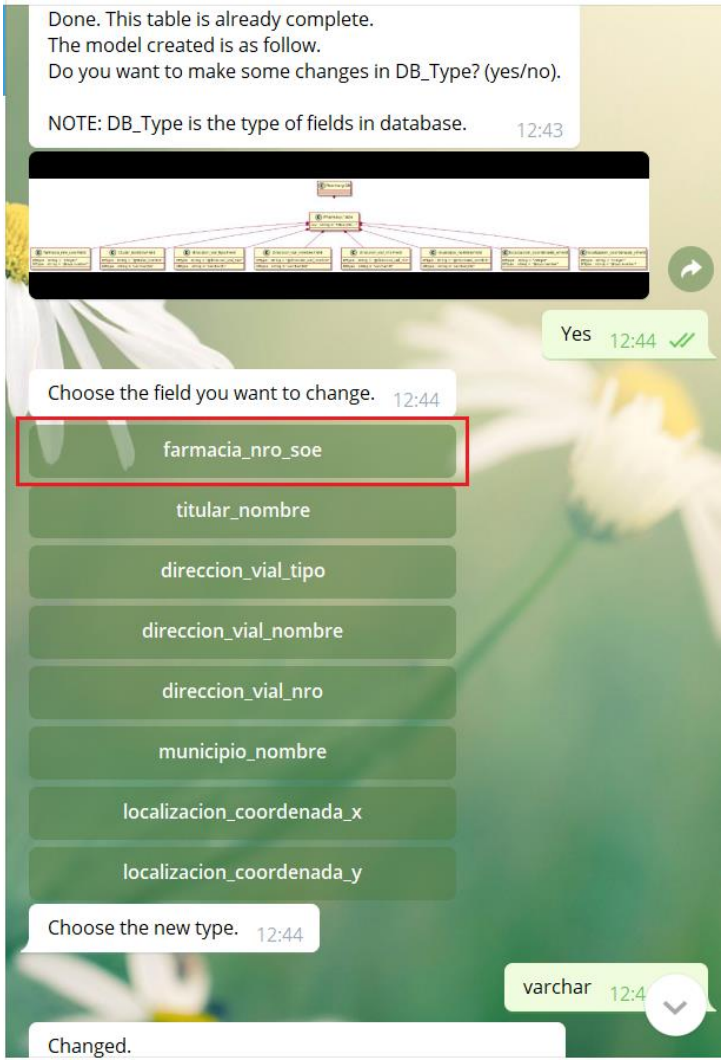


Figura 36: Cambio de tipo de la base de datos del campo *farmacia_nro_soe*

En concreto, el campo *farmacia_nro_soe* es un campo del tipo integer en la base de datos y, a modo de prueba, se procede a cambiarlo a varchar. En la figura 37 se puede ver el antes y en la 38 el después del cambio de tipo de *farmacia_nro_soe*. Por otro lado, en la figura 27 se visualizan las distintas opciones que tiene el usuario para elegir, es decir, los tipos de la base de datos a elegir.

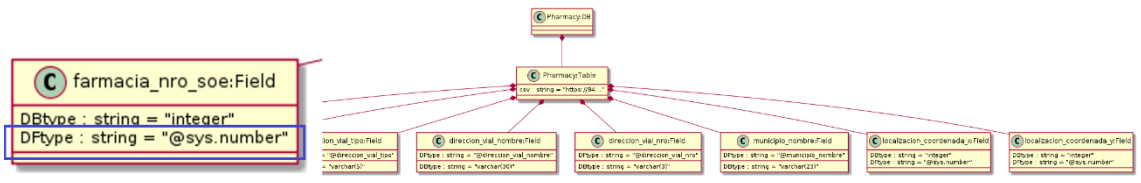


Figura 37: Antes del cambio: Tipos inferidos de cada campo para las columnas de la base de datos y las entidades de Dialogflow

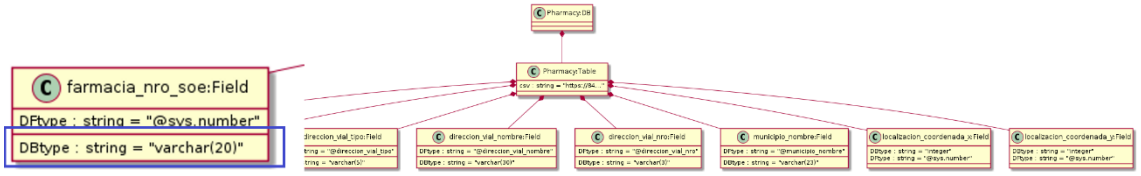


Figura 38: Después del cambio: Tipos de cada campo para las columnas de la base de datos y las entidades de Dialogflow

En este punto, GENBOT nos proporciona una url para descargar un zip que contiene un agente de Dialogflow preconstruido (figura 24). Los pasos siguientes a la descarga son los siguientes: (1) creación e importación del agente en Dialogflow, (2) creación de un bot en Telegram y (3) enlace del bot de Telegram con el agente de Dialogflow. Estos puntos se describen en mayor detalle en el Anexo I Crear un bot en Telegram y Anexo II Crear e importar un agente en Dialogflow.

Finalmente, una vez el chatbot está desplegado, se puede comenzar a hablar con él (figura 30). En este punto, nos proponemos visualizar la localización de una farmacia determinada. En concreto, la figura 34 muestra el momento en el que se le pide a éste visualizar dónde está la farmacia de *Mercedes Valdés Galera* representa. Como respuesta, el bot enviará una imagen señalando el punto en el mapa de la localización que pidió el usuario y, por otro lado, se envía una url que estará activa un día en la que se podrá acceder a un mapa interactivo.

Por otro lado, queremos visualizar la relación entre las variables de latitud y longitud en un gráfico. Para ello decidimos optar por un gráfico de dispersión para las variables de longitud y localización (figura 39).

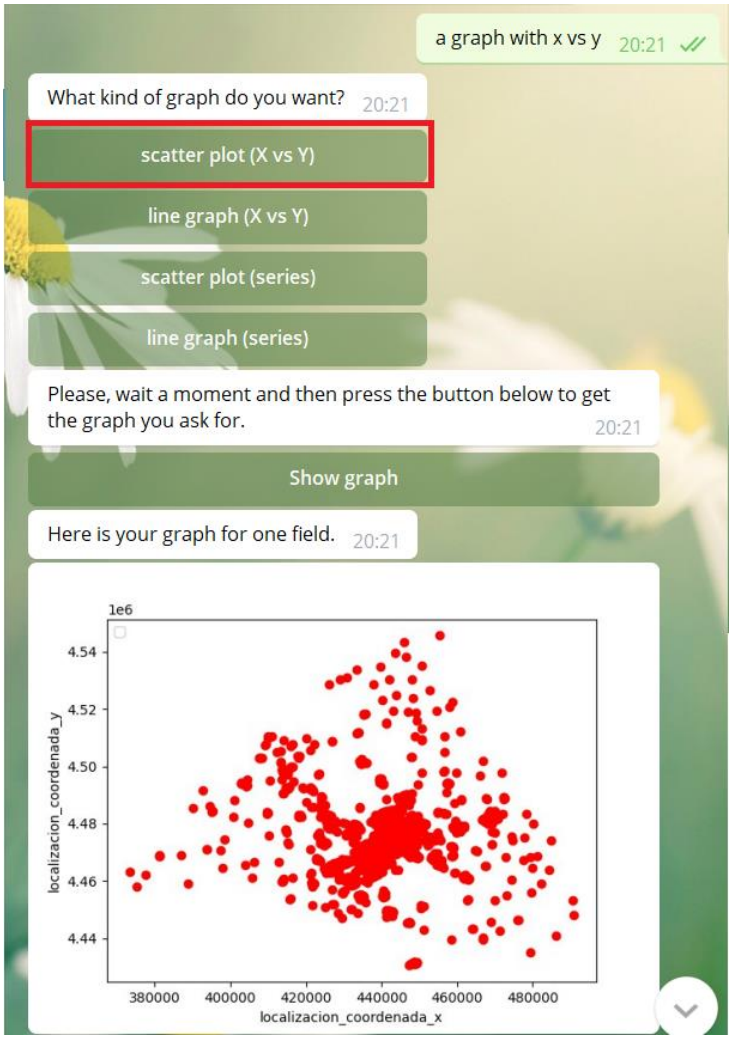


Figura 39: Gráfico de dispersión. *Localización_ coordenada_ y* vs *localización_ coordenada_ x*

Finalmente, se quiere ver de forma gráfica todos los valores que toma la variable *dirección_ vial_ tipo* se pide al chatbot por un gráfico de tarta (figura 33).

5.2 Caso de uso 2: Consultas

En este caso de uso se probará realizar consultas en lenguaje natural. Para esta ocasión se ha usado un dataset de la comunidad de Madrid que registra todos los libros prestados en las bibliotecas madrileñas.

A continuación, en la figura 40, se muestra la información que proporciona la comunidad de Madrid sobre este dataset.

Columna	Descripción	Tipo/Tamaño
prbarc	Código de barras del ejemplar	Numérico 10
prprsu	Código de sucursal del préstamo(1)	Alfanumérico 6
prcolp	Código de tipo de lector (2)	Alfanumérico 3
prcocp	Código de tipo de ejemplar (3)	Alfanumérico 3
prlebi	Biblioteca del lector	Alfanumérico 6
prlesu	Sucursal del lector (4)	Alfanumérico 6
pradul	Adulto / No adulto (1 / 0) (5)	Numérico 1
prfpre	Fecha del préstamo (6)	Fecha/Hora 24
prcocs	Soporte (7)	Alfanumérico 3
tititu	Título del libro (8)	Alfanumérico 60
tiauto	Autor	Alfanumérico

Figura 40: Descripción de las columnas del dataset por la comunidad de Madrid

Las consultas que se deciden que se realizarán en lenguaje natural con este bot son las siguientes:

1. Obtener todos los códigos de lector de todos los libros prestados.
2. Obtener todos los títulos de los libros prestados en una fecha y sucursal determinada.
3. Obtener todos los títulos de los libros prestados a usuarios con un determinado código de lector.

A continuación, podemos ver en la tabla 7 toda la información que hay que introducir en el archivo *properties* ubicado en la raíz del proyecto. En la columna de la derecha están las frases que se introducirían en el archivo *properties* y, en la columna de la izquierda, la sentencia correspondiente en lenguaje natural.

Lenguaje natural	Archivo properties
All code readers	All %prcolp:select%
Books loaned on 7th of may at the branch office with code BCDIC	%tititu:select% loaned on &pfrpre& at the branch office with code &prlesu&
All books loaned with reader code PRO	All %tititu:select% loaned with reader code &prcolp&

Tabla 7: Frases del archivo properties

Después, es necesario hablar con GENBOT para generar el chatbot requerido. La conversación es muy similar a la que se describe en el primer caso de uso. Una vez finalizada, se obtiene el zip con el agente que es necesario exportar a Dialogflow y conectar con un bot de Telegram.

A continuación, se muestran los sinónimos que se han introducido para cada campo de la base de datos (tabla 8). En este dataset son importantes los sinónimos ya que los campos no tienen nombres descriptivos. De esta forma, las consultas podrán realizarse en un lenguaje simple.

Nombre campo	Sinónimos
prbarc	barcode
prprsu	prprsu (No se han dado sinónimos ya que solo tiene un valor y puede confundirse con prlesu. Se ha evitado usar)
prcolp	reader type, reader
prcocp	book type
prlebi	Library
prlesu	branch office, office, branch
pradul	adult
prfpre	date
prcocs	book format
tititu	title, book title
tiauto	author

Tabla 8: Sinónimos campos tabla *library*

Una vez completado el archivo properties, finalizada la conversación con GENBOT y desplegado el bot resultante, se puede empezar a hablar con el chatbot a través de Telegram.

La primera consulta hace referencia a una consulta simple, sin filtros, en la que se quiere visualizar todos los códigos de lector de la base de datos. La figura 9 muestra el resultado del bot donde el usuario introduce un mensaje asociado a esta consulta.

Por otro lado, la segunda consulta requiere un filtro de dos parámetros: fecha y sucursal del lector. En este caso, la figura 10, al igual que la anterior, muestra distintas consultas y el resultado que genera el bot en consecuencia. Este resultado es el mismo para ambas consultas aun habiendo cambiado el formato de la fecha.

Dialogflow es capaz de reconocer, gracias a la entidad fecha, distintos formatos de fecha. A modo de ejemplo, se ha sacado una captura de pantalla del chat de prueba de Dialogflow para esta consulta ilustrado en la figura 42. En ella se ve cómo dialogflow reconoce la fecha y la convierte en un tipo predefinido final de la entidad *sys.date.*

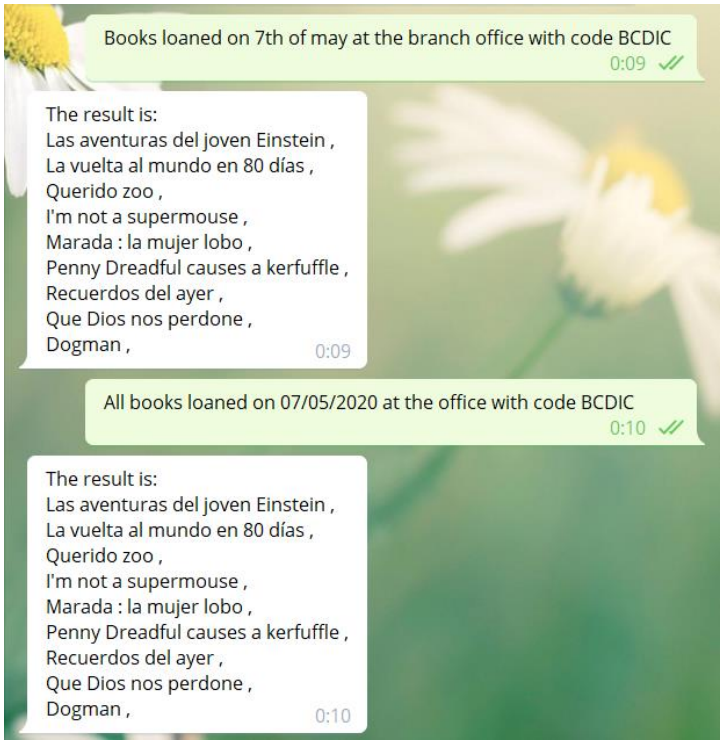


Figura 41: Consultas "obtener todos los libros prestados un determinado día en una determinada sucursal"

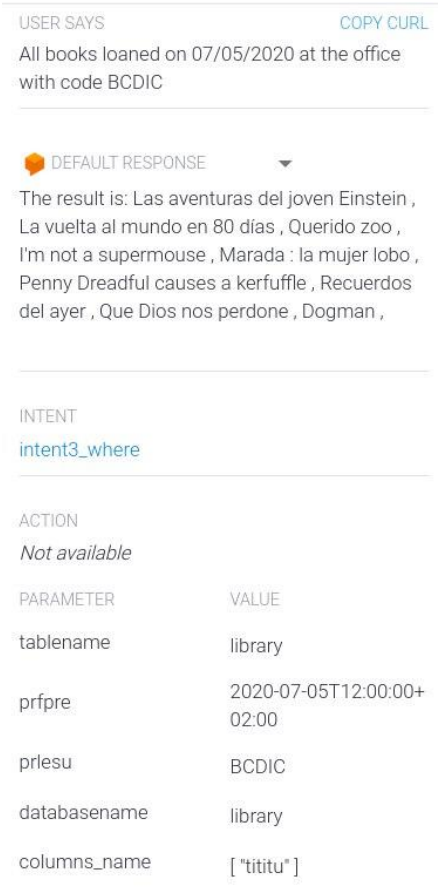


Figura 42: Chat de pruebas de Dialogflow

Finalmente, la tercera consulta requiere de sólo un campo por el que filtrar, es muy parecida a la segunda pero más sencilla. La figura 10 muestra un ejemplo de

cómo un usuario realizaría esta consulta. En este caso, el número de registros es superior al del resto de consultas y es necesario incluir los resultados en un fichero .txt que es enviado al usuario a través de una url.

5.3 Conclusiones

Tal como se ha visto en los casos de uso de los apartados anteriores, el sistema es capaz de responder ante distintas consultas a la fuente de datos. Además, es posible generar y visualizar gráficos de distintos tipos. Por último, también se puede conocer la ubicación mediante mapas visuales e interactivos de distintas ubicaciones pedidas por el usuario.

En cuanto a los aspectos negativos, la fluidez de uso con el sistema se ve interrumpida por ciertas acciones más técnicas que son necesarias que el usuario realice para desplegar completamente los chatbots autogenerados (ver Anexo I Crear un bot en Telegram y Anexo II Crear e importar un agente en Dialogflow y el apartado 6.2 de limitaciones). Por el contrario, los mapas y los gráficos dotan al sistema de riqueza visual y lo diferencian. Además, poder realizar consultas en lenguaje natural abre muchas posibilidades a la hora de obtener información de una fuente de datos por cualquier tipo de usuario.

Además, el nivel de comprensión de cada bot queda ligado a la complejidad del dataset y de las consultas especificadas en el archivo *properties.* Un error común en estos chatbots es el no diferenciar correctamente entre aquellos nombres de columnas que hacen referencia a los datos que quiere obtener el usuario y los nombres de columnas de los filtros.

Capítulo 6
Conclusiones y trabajo futuro

En el presente capítulo se exponen las conclusiones a las que se han llegado con la realización de este proyecto (Sección 3), las limitaciones (Sección 6.2) y el trabajo futuro (Sección 6.3).

6.1 Conclusiones del Proyecto

La herramienta que se propone en este documento se llama GENBOT, un chatbot integrado en Telegram con el cual se interactúa con lenguaje natural. Su objetivo es el de crear chatbots específicos de consulta y visualización de datos para fuentes de datos abiertas en formato csv.

Los chatbots generados por GENBOT están creados para su uso por usuarios inexpertos en el tratamiento y consulta de datos. Éstos pueden mantener una conversación mediante lenguaje natural para consultar y visualizar información de una fuente de datos concreta sin necesidad de conocimientos técnicos.

A través de estos chatbots, los usuarios pueden realizar un conjunto establecido de consultas a la fuente de datos en lenguaje natural. Los tipos de consultas válidos que los chatbots reconocerán son los definidos al interactuar con GENBOT.

Por otro lado, los chatbots generados también podrán mostrar mapas y gráficas para visualizar la información de las columnas de la fuente de datos.

Debido a las limitaciones número 1 y 2, descritas en el 6.2, GENBOT no despliega automáticamente un chatbot específico funcional. El resultado de la interacción con GENBOT es un agente de Dialogflow preconstruido que hay que importar a Dialogflow y vincularlo con un bot de Telegram. Este hecho puede complicar el uso de GENBOT, sin embargo, para solventar la dificultad que pueden añadir estas limitaciones, se añaden los anexos I y II a este documento con explicaciones paso a paso tanto para importar el agente a Dialogflow como para crear un bot en Telegram.

6.2 Limitaciones

El diseño del sistema se ha visto afectado por las limitaciones que se describen en los párrafos siguientes.

No se pueden desplegar agentes de Dialogflow de manera automática, es necesario crearlos manualmente a través de la consola de Dialogflow.

Al igual que ocurre con los agentes y Dialogflow, tampoco está permitido desplegar bots de Telegram de manera automática, siendo necesario crearlos a través del bot de Telegram llamado BotFather.

No se pueden enviar archivos desde Telegram (como el csv de la fuente de datos). Aunque Telegram cuenta con opción de envío de archivos, esta limitación proviene de Dialogflow, al integrar el agente de manera autómatica a través de "Integrations" con Telegram.

Los timeouts fijados por Dialogflow no pueden modificarse.

6.3 Trabajos futuros

Los puntos en los que se puede trabajar para mejorar y completar el proyecto son los siguientes.

Mejorar el modelo de Inteligencia Artificial de Dialogflow con mayor número de frases de ejemplo para cada uno de los intents que se han definido para crear el sistema y ampliar el uso de otros idiomas distintos al inglés.

Añadir al sistema la capacidad de proporcionar un agente preconstruido en Dialogflow ya integrado con un determinado bot de telegram. En el caso de añadir esta nueva funcionalidad será preciso proteger de manera adecuada el token del bot de Telegram al que se quiera enganchar el agente de Dialogflow.

Ampliar los formatos soportados, más allá del CSV, para trabajar con más fuentes de datos abiertos.

Añadir consultas más sofisticadas, por ejemplo, permitir consultas que requieran uso de funciones de SQL como *min, máx etc.* Además, mejorar la especificación de éstas en el sistema (archivo *properties*).

Finalmente, crear una guía y un servicio de despliegue automático de la herramienta para replicaciones.

Referencias

[1] Generalitat de Catalunya, «¿Qué son los datos abiertos?,» [En línea]. Available: http://governobert.gencat.cat/es/dades_obertes/que-son-les-dades-obertes/. [Último acceso: 18 Noviembre 2019].

[2] Open Data Commons, «Open Data Commons is the home of a set of legal tools to help you provide and use Open Data,» Open Data Commons, [En línea]. Available: https://www.opendatacommons.org/. [Último acceso: 05 mayo 2020].

[3] Open Knowledge Foundation, «Open Data HandBook,» [En línea]. Available: https://opendatahandbook.org/. [Último acceso: 18 noviembre 2019].

[4] John M. Carroll, «The encyclopedia of human-computer interaction,» Interaction Design Foundation, [En línea]. Available: https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/human-computer-interaction-brief-intro. [Último acceso: 20 noviembre 2019].

[5] Lorenz Cuno Klopfenstein, Saverio Delpriori, et atl., «The Rise of Bots: A Survey of Conversational Interfaces, Patterns, and Paradigms,» *DIS '17 Proceedings of the 2017 Conference on Designing Interactive Systems ,* pp. 555-565 , 10 junio 2017.

[6] Abhishek Singh, Karthik Ramasubramanian, Shrey Shi, Building an Enterprise Chatbot, Apress, Berkeley, CA, 2019, p. 35.

[7] XOXCO, «Botkit,» [En línea]. Available: https://botkit.ai/. [Último acceso: 19 noviembre 2019].

[8] FLG Software Ltd, «Flow XO: AI Online Chatbot Software, Live Chat on Websites,» [En línea]. Available: https://flowxo.com/. [Último acceso: 19 noviembre 2019].

[9] chatfuel, «Chatfuel,» [En línea]. Available: https://chatfuel.com. [Último acceso: 19 noviembre 2019].

[10] Recime Inc., «Smartlopp,» [En línea]. Available: https://smartloop.ai/. [Último acceso: 19 noviembre 2019].

[11] HELLO UMI SL, «Intuitive Conversational Chatbot Builder | Landbot.io,» [En línea]. Available: https://landbot.io. [Último acceso: 19 noviembre 2019].

[12] Google, «Dialogflow,» [En línea]. Available: https://dialogflow.com/. [Último acceso: 19 noviembre 2019].

[13] Amazon, «Amazon Lex – Build Conversation Bots - Amazon Web Services,» [En línea]. Available: https://aws.amazon.com/es/lex/. [Último acceso: 19 noviembre 2019].

[14] Amazon, «Características de AWS Lambda,» Amazon, [En línea]. Available: https://aws.amazon.com/es/lambda/features/. [Último acceso: 09 05 2020].

[15] Amazon, «Amazon Web Services,» Amazon, [En línea]. Available: https://aws.amazon.com/es/. [Último acceso: 09 05 2020].

[16] IBM, «IBM Watson | IBM,» [En línea]. Available: https://www.ibm.com/watson. [Último acceso: 20 noviembre 2019].

[17] Microsoft, «Language Understanding (LUIS),» [En línea]. Available: https://www.luis.ai/home. [Último acceso: 20 noviembre 2019].

[18] J. Lana, «Blogthinkbig.com,» 20 abril 2016. [En línea]. Available: https://blogthinkbig.com/chat-bots-o-conversaciones-automaticas-a-traves-de-aplicaciones-de-mensajeria-movil. [Último acceso: 30 octubre 2019].

[19] Whatsapp Inc., «Search Results,» [En línea]. Available: https://www.whatsapp.com/. [Último acceso: 21 noviembre 2019].

[20] Wikipedia, «WhatsApp,» [En línea]. Available: https://es.wikipedia.org/wiki/WhatsApp#Esquema_de_precios. [Último acceso: 21 noviembre 2019].

[21] Tole Sutikno, Lina Handayani, et al., «WhatsApp, Viber and Telegram: which is the Best for Instant,» *International Journal of Electrical and Computer Engineering (IJECE),* vol. 6, nº 3, p. 909~914, 2016.

[22] Facebook, «Facebook,» [En línea]. Available: https://www.facebook.com. [Último acceso: 20 noviembre 2019].

[23] J. Facchin, «¿Qué es Facebook, para qué sirve y cómo funciona la mayor plataforma social del mundo?,» [En línea]. Available: https://josefacchin.com/facebook-que-es-como-funciona/. [Último acceso: 21 noviembre 2019].

[24] Faceboock, «Messenger,» [En línea]. Available: https://www.messenger.com/. [Último acceso: 21 noviembre 2019].

[25] Navid Yaghmazadeh, Yuepeng Wang, et al., «SQLizer: query synthesis from natural language,» *Proceedings of the ACM on Programming Languages,* vol. Volume 1 Issue OOPSLA, p. Article No. 63 , Octubre 2017.

[26] Jonathan Berant, Andrew Chou, et al., «Semantic Parsing on Freebase from Question-Answer Pairs,» vol. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, nº D13-1160, p. 1533–1544, Octubre 2013.

[27] Christopher Manning, Mihai Surdeanu, et al., «The Stanford CoreNLP Natural Language Processing Toolkit,» vol. Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, nº P14-5010, p. 55–60, Junio 2014.

[28] Victor Zhong, Caiming Xiong, Richard Socher, «Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning,» *arXiv preprint arXiv:1709.00103,* 31 Agosto 2017.

[29] P. Rezaei, «nl2sql,» [En línea]. Available: https://github.com/prezaei85/nl2sql. [Último acceso: 21 noviembre 2019].

[30] M. Grinberg, Flask web development: developing web applications with python, O'Reilly Media, Inc., 2018.

[31] John Hunter, Darren Dale, et al., «Matplotlib,» The Matplotlib development team, [En línea]. Available: https://matplotlib.org/. [Último acceso: 04 junio 2020].

[32] Mapbox team, «mapbox,» [En línea]. Available: https://www.mapbox.com/. [Último acceso: 26 Julio 2020].

[33] Folium team, «folium,» [En línea]. Available: https://github.com/python-visualization/folium. [Último acceso: 26 julio 2020].

[34] Vladimir Agafonkin & Leaflet team, «leafletjs,» [En línea]. Available: https://leafletjs.com/. [Último acceso: 26 julio 2020].

[35] Rasa Team Inc, «Rasa: Open source conversational AI - Rasa,» [En línea]. Available: https://rasa.com. [Último acceso: 19 noviembre 2019].

[36] M. JC, «The Open Knowledge Foundation: Open Data Means Better Science,» *PLoS Biol,* vol. 12, nº 9, 2011.

[37] Nigel Shadbolt, Kieron O'Hara, et al., «Linked open government data: lessons from Data.gov.uk,» *IEEE Intelligent Systems,* vol. 27, nº 3, pp. 16-24, 2012.

[38] I. Encyclopædia Britannica, «SQL,» [En línea]. Available: https://www.britannica.com/technology/database. [Último acceso: 22 noviembre 2019].

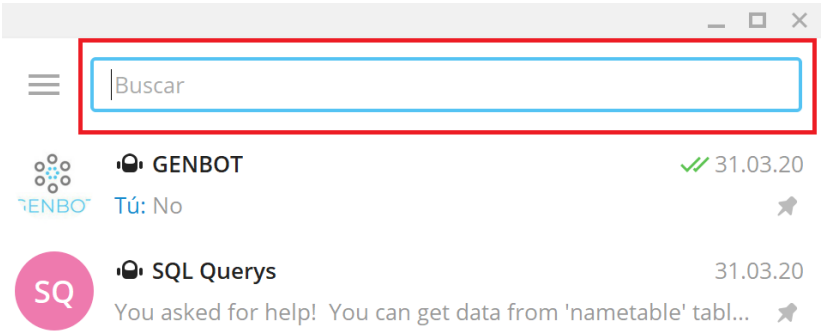
[39] R. F. B. Donald D. Chamberlin, «SEQUEL: A structured English query language,» de *SIGFIDET '74: Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control*, Ann Arbor, Michigan, Association for Computing Machinery, 1974, p. 249–264.

Anexo I
Crear un bot en Telegram

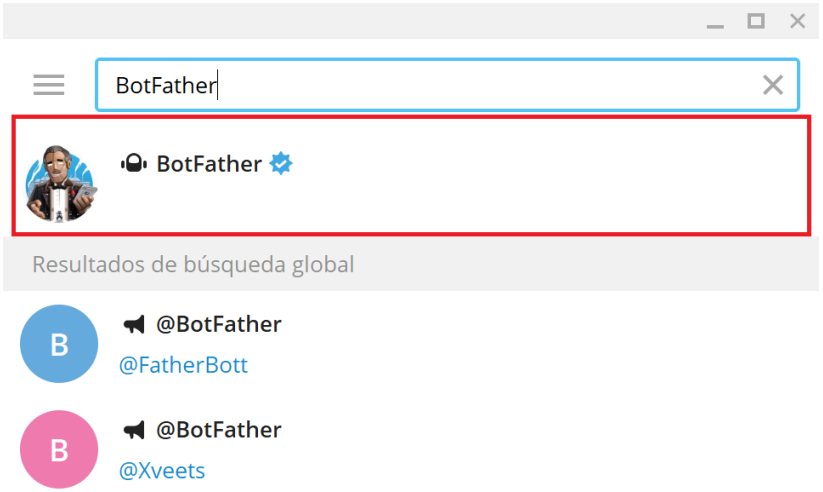
En este anexo se enseña de forma gráfica y detallada cómo crear un bot en Telegram.

1. Abrir Telegram (válido cualquier dispositivos que tenga instalado Telegram).

2. Abrir una conversación con BotFather.
   a. Dirigirse al cuadro de búsqueda de Telegram.

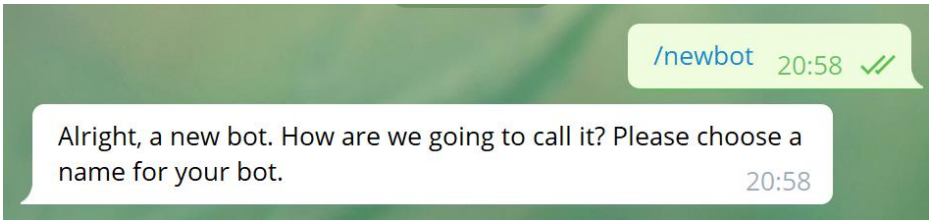


   b. Introducir "BotFather" y seleccionar a BotFather (la primera opción) para iniciar una conversación con él.



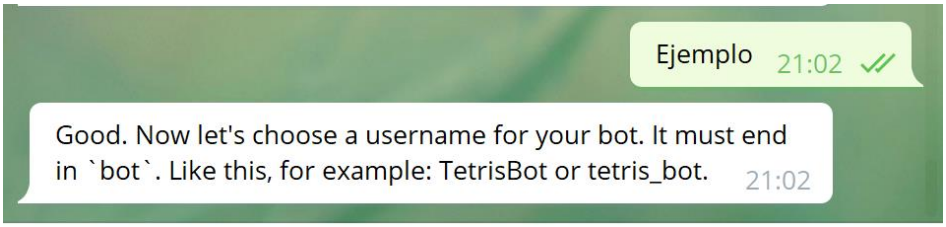
Al pulsar en BotFather, Telegram tiene que abrir una conversación con él.

3. Hablar con BotFather y proporcionarle toda la información que pida para crear el nuevo bot.
    a. Escribir "newbot" o desplegar la lista de comandos y hacer clic en "newbot". Este comando inicia una conversación para crear un nuevo bot.

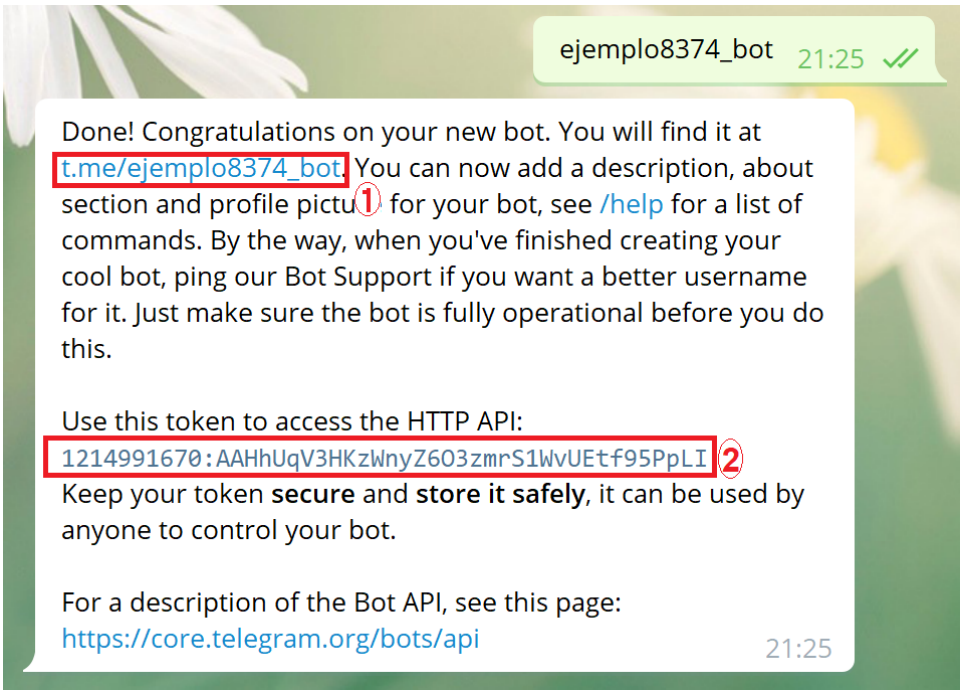


    b. BotFather pide que se le proporcione un nombre para el bot que se está creando.

Escribir el nombre que más guste y enviar.



c. Ahora, hay que eligir un nick para el bot y enviarlo. El nick tiene que acabar en "bot" y tiene que ser único. Si ya existe otro bot con el mismo nick, BotFather te lo hará saber para que elijas otro.



¡El bot ya ha sido creado! Se puede empezar a hablar con él si se pulsa el link del cuadro número 1 de la imagen anterior o si se busca por su nick a través del buscador de Telegram usado en el paso 2.a.

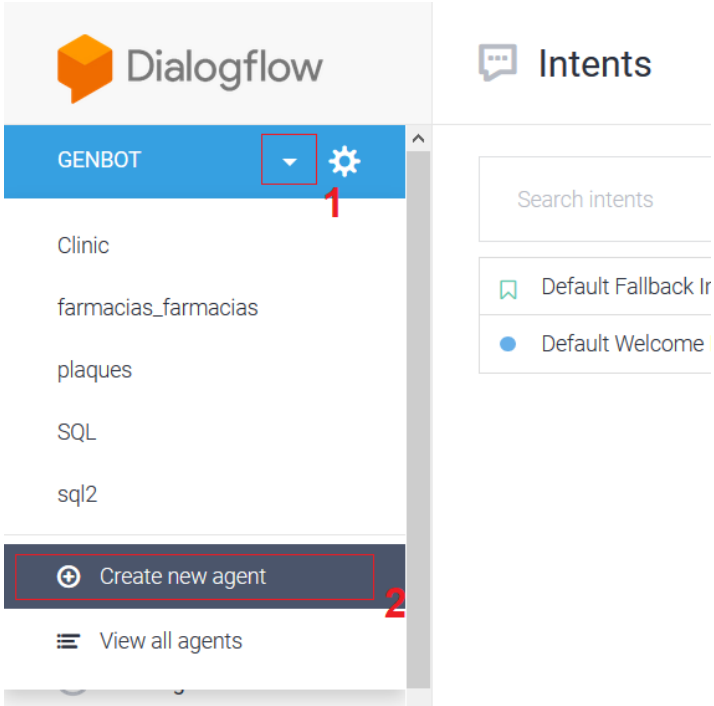
El cuadro número 2 señala el token necesario para acceder al nuevo bot creado a través de la API HTTP de Telegram. Es un código privado que permitirá acceder al bot para hablarle, modificarle, leer mensajes o enlazarlo con otros servicios. Se debe guardar.

Anexo II
Crear e importar un agente en Dialogflow

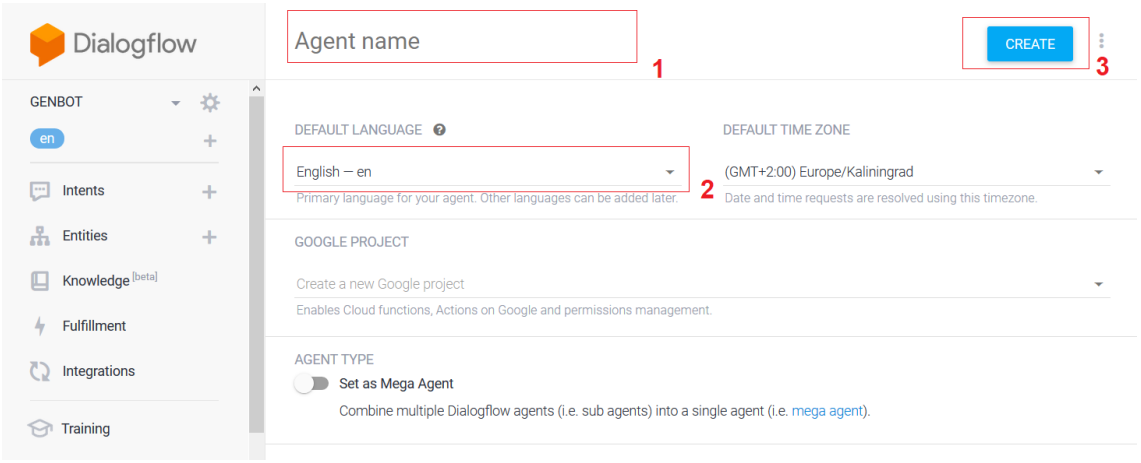
En este anexo se mostrará cómo importar a Dialgflow los agentes preconstruidos que genera GENBOT. También se ilustrará como conectarlo con Telegram.

1. Registro/inicio de sesión en Dialogflow.
    a. Entrar en https://dialogflow.cloud.google.com y pulsar registrar.
    b. Iniciar sesión con una cuenta de Gmail y aceptar el acceso de Dialogflow a la cuenta de Gmail en caso de acceder por primera vez.
    c. Una vez iniciada la sesión se accederá a la consola de Dialogflow.

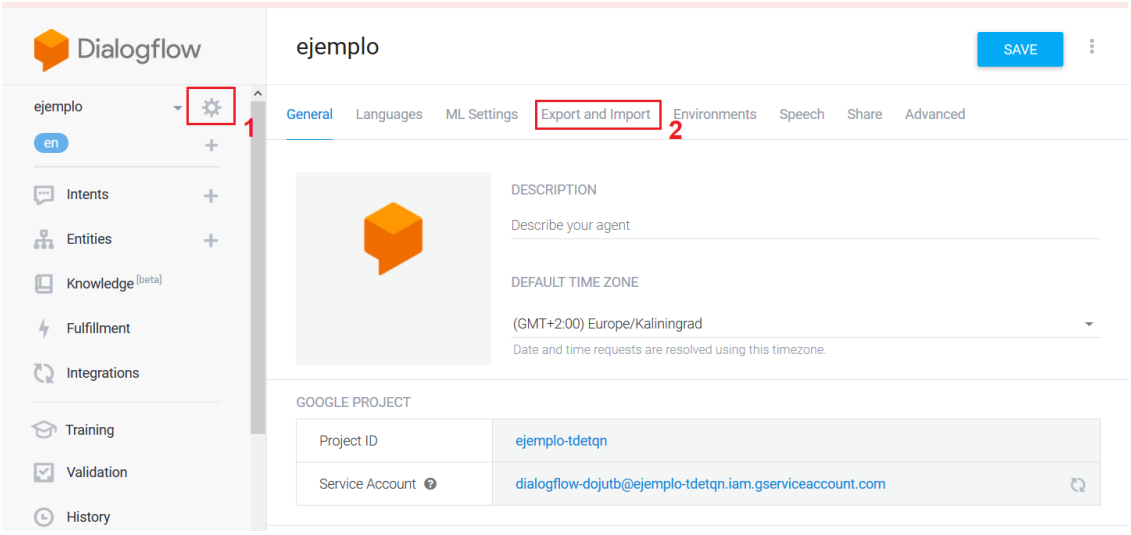
2. Crear agente de Dialogflow.
    a. Desplegar la lista de agente (1) y elegir "Create new agent" (2).



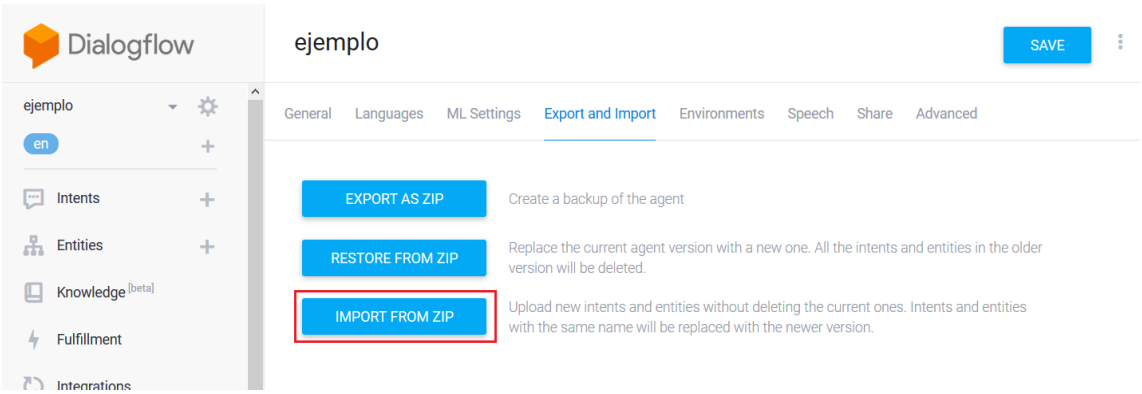
b. Elegir un nombre para el agente (1), un idioma (2) y pulsar "CREATE" (3). En esta pantalla también se pueden elegir otras características opcionales como la zona horaria y el proyecto al que pertenecerá el agente (por defecto se crea uno nuevo).



3. Importar el agente preconstruido que genera GENBOT.
   a. Acceder a la configuración del agente (1) y después a "Export and Import" (2).

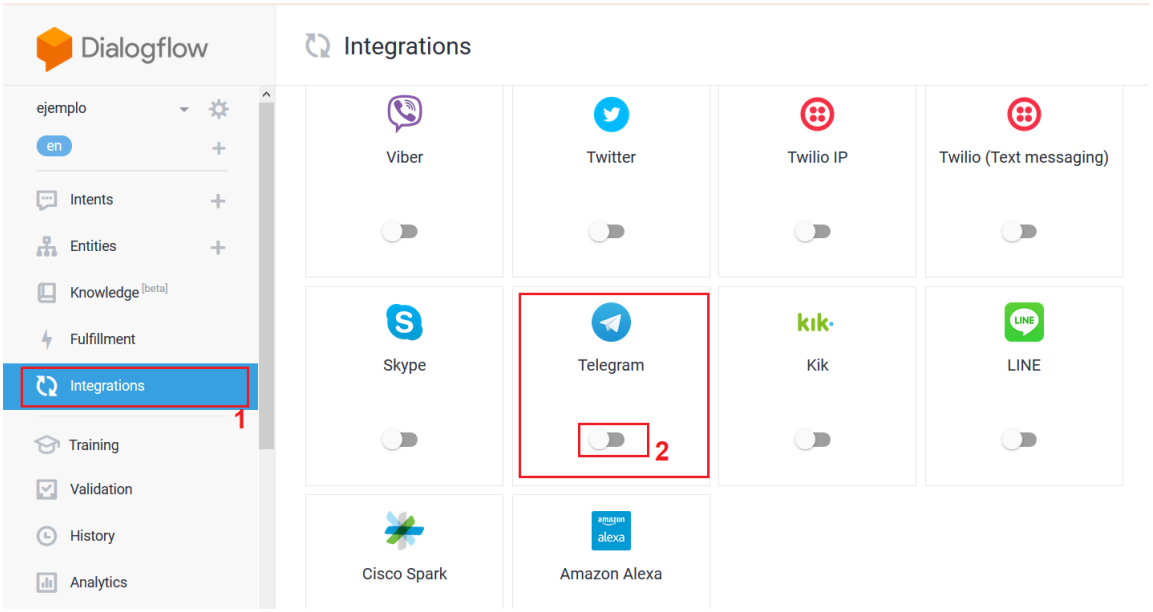


b. Dentro de la sección “Export and Import” seleccionar “Import from zip”.

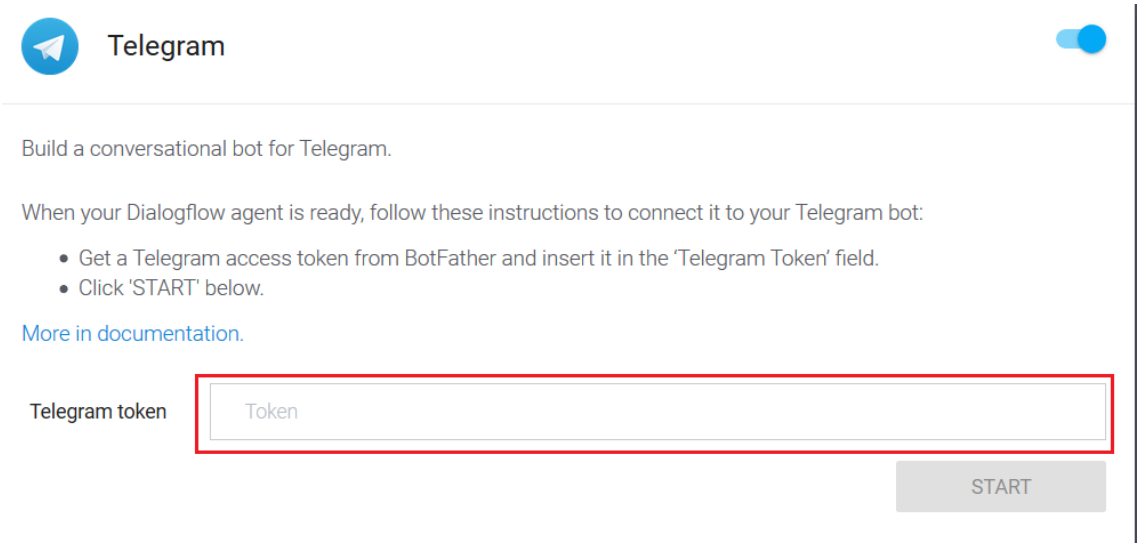


c. Elegir el zip generado por GENBOT y aceptar. Esperar a que el agente se suba y se entrene.

4. Integrar agente con bot en Telegram.
   a. Acceder a la sección de integraciones (1) y buscar y activar la integración con Telegram (2).



b. Ingresar el token del bot de Telegram al que queramos enganchar el agente de Dialogflow.



Ya está listo el agente que nos proporciona GENBOT para hablar con él a través del bot de Telegram al que le hayamos conectado.